EVALITA
Evaluation of NLP and Speech Tools for Italian

Proceedings of the Final Workshop

7 December 2016, Naples

Editors:

Pierpaolo Basile
Franco Cutugno
Malvina Nissim
Viviana Patti
Rachele Sprugnoli

aA

EVALITA. Evaluation of NLP and Speech Tools for Italian

Proceedings of the Final Workshop 7 December 2016, Naples

Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti and Rachele Sprugnoli (dir.)

DOI: 10.4000/books.aaccademia.1899
Publisher: Accademia University Press
Place of publication: Torino
Year of publication: 2016
Published on OpenEdition Books: 28 August 2017
Serie: Collana dell'Associazione Italiana di Linguistica Computazionale
Electronic ISBN: 9788899982553

Printed version
Number of pages: 218

http://books.openedition.org

Electronic reference
BASILE, Pierpaolo (ed.) ; et al. *EVALITA. Evaluation of NLP and Speech Tools for Italian: Proceedings of the Final Workshop 7 December 2016, Naples.* New edition [online]. Torino: Accademia University Press, 2016 (generated 30 August 2017). Available on the Internet: <http://books.openedition.org/aaccademia/1899>. ISBN: 9788899982553. DOI: 10.4000/books.aaccademia.1899.

EVALITA
Evaluation of NLP and Speech Tools for Italian

Proceedings of the Final Workshop

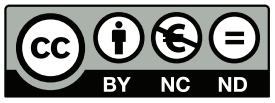
7 December 2016, Naples

Editors:

Pierpaolo Basile
Franco Cutugno
Malvina Nissim
Viviana Patti
Rachele Sprugnoli

aA Accademia university press

aA



Accademia University Press
via Carlo Alberto 55
I-10123 Torino
info@aAccademia.it

isbn 978-88-99982-09-6
www.aAccademia.it/EVALITA_2016

Preface to the EVALITA 2016 Proceedings*

EVALITA is the evaluation campaign of Natural Language Processing and Speech Tools for the Italian language: since 2007 shared tasks have been proposed covering the analysis of both written and spoken language with the aim of enhancing the development and dissemination of resources and technologies for Italian. EVALITA is an initiative of the Italian Association for Computational Linguistics (AILC, http://www.ai-lc.it/) and it is supported by the NLP Special Interest Group of the Italian Association for Artificial Intelligence (AI*IA, http://www.aixia.it/) and by the Italian Association of Speech Science (AISV, http://www.aisv.it/).

In this volume, we collect the reports of the tasks' organisers and of the participants to all of the EVALITA 2016's tasks, which are the following: ArtiPhone - *Articulatory Phone Recognition*; FactA - *Event Factuality Annotation*; NEEL-IT - *Named Entity rEcognition and Linking in Italian Tweets*; PoSTWITA - *POS tagging for Italian Social Media Texts*; QA4FAQ - *Question Answering for Frequently Asked Questions*; SENTIPOLC - *SENTIment POLarity Classification*. Notice that the volume does not include reports related to the *IBM Watson Services Challenge* organised by IBM Italy, but information can be found at http://www.evalita.it/2016/tasks/ibm-challenge. Before the task and participant reports, we also include an overview to the campaign that describes the tasks in more detail, provides figures on the participants, and, especially, highlights the innovations introduced at this year's edition. An additional report presents a reflection on the outcome of two questionnaires filled by past participants and organisers of EVALITA, and of the panel "Raising Interest and Collecting Suggestions on the EVALITA Evaluation Campaign" held at CLIC-it 2015. The abstract of Walter Daelemans's invited talk is also included in this volume.

The final workshop was held in Naples on the 7th of December 2016 as a co-located event of the Third Italian Conference on Computational Linguistics (CLiC-it 2016, http://clic-it2016.dieti.unina.it/). During the workshop, organisers and participants, affiliated both to academic institutions and industrial companies, disseminated the results of the evaluation and the details of the developed systems through oral and poster presentations.

We thank all the people and institutions involved in the organisation of the tasks, and all participating teams, who contributed to the success of the event. A special thank is also due to AILC, which is for the first time the official mother of EVALITA. We are especially grateful to AILC not only for its support during the whole of the campaign's organisation, but also for having financially contributed to data creation for the SENTIPOLC task, by funding the crowdsourcing experiment. Thanks are also due to AI*IA and AISV for endorsing EVALITA, and to FBK for making the web platform available once more for this edition (http://www.evalita.it).

Last but not least, we heartily thank our invited speaker, Walter Daelemans from the University of Antwerp, Belgium, for agreeing to share his expertise on key topics of EVALITA 2016.

November 2016

EVALITA 2016 Co-Chairs

Pierpaolo Basile
Franco Cutugno
Malvina Nissim
Viviana Patti
Rachele Sprugnoli

*Originally published online by CEUR Workshop Proceedings (CEUR-WS.org, ISSN 1613-0073)

Chairs

Pierpaolo Basile, University of Bari, Italy
Franco Cutugno, University of Naples “Federico II”, Italy
Malvina Nissim, University of Groningen, The Netherlands
Viviana Patti, University of Turin, Italy
Rachele Sprugnoli, Fondazione Bruno Kessler and University of Trento, Italy

Steering Committee

Leonardo Badino, Istituto Italiano di Tecnologia, Italy
Francesco Barbieri, Universitat Pompeu Fabra, Spain
Pierpaolo Basile, University of Bari “Aldo Moro”, Italy
Valerio Basile, Université Côte d’Azur, Inria, CNRS, France
Andrea Bolioli, CELI s.r.l - Language Technology, Italy
Cristina Bosco, University of Turin, Italy
Annalina Caputo, ADAPT Centre Dublin, Ireland
Tommaso Caselli, Vrije Universiteit Amsterdam, The Netherlands
Danilo Croce, University of Rome “Tor Vergata”, Italy
Franco Cutugno, University of Naples “Federico II”, italy
Marco De Gemmis, University of Bari, Italy
Anna Lisa Gentile, University of Mannheim, Germany
Bertrand Higy, Istituto Italiano di Tecnologia, Italy
Pietro Leo, IBM GBS BAO Advanced Analytics Services and MBLab, Italy
Pasquale Lops, University of Bari “Aldo Moro”, Italy
Francesco Lovecchio, AQP s.p.a., Italy
Vito Manzari, SudSistemi s.r.l., Italy
Alessandro Mazzei, University of Turin, Italy
Anne-Lyse Minard, Fondazione Bruno Kessler, Italy
Malvina Nissim, University of Groningen, The Netherlands
Nicole Novielli, University of Bari “Aldo Moro”, Italy
Viviana Patti, University of Turin, Italy
Giuseppe Rizzo, Istituto Superiore Mario Boella, Italy
Manuela Speranza, Fondazione Bruno Kessler, Italy
Rachele Sprugnoli, Fondazione Bruno Kessler and University of Trento, Italy
Fabio Tamburini, University of Bologna, Italy

Contents

PART I

INTRODUCTION TO EVALITA 2016

EVALITA 2016: Overview of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian

Pierpaolo Basile
University of Bari
Via E.Orabona, 4
70126 Bari, Italy
basilepp@di.uniba.it

Franco Cutugno
University Federico II
Via Claudio 21
80126 Naples, Italy
cutugno@unina.it

Malvina Nissim
University of Groningen
Oude Kijk in t Jatstraat 26
9700 AS Groningen, NL
m.nissim@rug.nl

Viviana Patti
University of Turin
c.so Svizzera 185
I-10149 Torino, Italy
patti@di.unito.it

Rachele Sprugnoli
FBK and University of Trento
Via Sommarive
38123 Trento, Italy
sprugnoli@fbk.eu

1 Introduction

EVALITA[1] is the evaluation campaign of Natural Language Processing and Speech Tools for the Italian language. The aim of the campaign is to improve and support the development and dissemination of resources and technologies for Italian. Indeed, many shared tasks, covering the analysis of both written and spoken language at various levels of processing, have been proposed within EVALITA since its first edition in 2007. EVALITA is an initiative of the Italian Association for Computational Linguistics[2] (AILC) and it is endorsed by the Italian Association of Speech Science[3] (AISV) and by the NLP Special Interest Group of the Italian Association for Artificial Intelligence[4] (AI*IA).

Following the success of the four previous editions, we organised EVALITA 2016 around a set of six shared tasks and an application challenge. In EVALITA 2016 several novelties were introduced on the basis of the outcome of two questionnaires and of the fruitful discussion that took place during the panel "Raising Interest and Collecting Suggestions on the EVALITA Evaluation Campaign" held in the context of the second Italian Computational Linguistics Conference (CLiC-it 2015) (Sprugnoli et al., 2016). Examples of these novelties are a greater involvement of industrial companies in the organisation of tasks, the introduction of a task and a challenge that are strongly application-oriented, and the creation of cross-task shared data. Also, a strong focus has been placed on using social media data, so as to promote the investigation into the portability and adaptation of existing tools, up to now mostly developed for the newswire domain.

2 Tasks and Challenge

As in previous editions, both the tasks and the final workshop were collectively organised by several researchers from the community working on Italian language resources and technologies. At this year's edition, the community organised six tasks and one additional challenge.

Standard Tasks Of the six tasks organised in the context of this year EVALITA, five dealt with different aspects of processing written language, with a specific focus on social media, and one on speech. A brief description of each task is given below.

- **ArtiPhon** – *Articulatory Phone Recognition*. In this task, participants had to build a speaker-dependent phone recognition system that is to be evaluated on mismatched speech rates. While training data consists of read speech where the speaker was required to keep a constant speech rate, testing data range from slow and hyper-articulated speech to fast and hypo-articulated speech (Badino, 2016).

[1] http://www.evalita.it
[2] http://www.ai-lc.it/
[3] http://www.aisv.it/
[4] http://www.aixia.it/

- **FactA** – *Event Factuality Annotation*. In this task, the factuality profiling of events is represented by means of three attributes associated to event mentions, namely: certainty, time, and polarity. Participating systems were required to provide the values for these three attributes (Minard et al., 2016).

- **NEEL-it** – *Named Entity rEcognition and Linking in Italian Tweets*. The task consists in automatically annotating each named entity mention (belonging to the following categories: Thing, Event, Character, Location, Organization, Person and Product) in a tweet by linking it to the DBpedia knowledge base (Basile et al., 2016).

- **PoSTWITA** – *POS tagging for Italian Social Media Texts*. The task consists in Part-Of-Speech tagging tweets, rather than more standard texts, that are provided in their already tokenised form (Bosco et al., 2016).

- **QA4FAQ** – *Question Answering for Frequently Asked Questions*. The goal of this task is to develop a system retrieving a list of relevant FAQs and corresponding answers related to a query issued by an user (Caputo et al., 2016).

- **SENTIPOLC** – *SENTIment POLarity Classification*. The task consists in automatically annotating tweets with a tuple of boolean values indicating the messages subjectivity, its polarity (positive or negative), and whether it is ironic or not (Barbieri et al., 2016).

Application Challenge In addition to the more standard tasks described above, for the first time EVALITA included a *challenge*, organised by IBM Italy. The **IBM Watson Services Challenge**'s aim is to create the most innovative app on Bluemix services[5], which leverages at least one Watson Service, with a specific focus on NLP and speech services for Italian (`http://www.evalita.it/2016/tasks/ibm-challenge`).

3 Participation

The tasks and the challenge of EVALITA 2016 attracted the interest of a large number of researchers, for a total of 96 single registrations. Overall, 34 teams composed of more than 60 individual participants from 10 different countries[6] submitted their results to one or more different tasks of the campaign.

A breakdown of the figures per task is shown in Table 1. With respect to the 2014 edition, we collected a significantly higher number of registrations (96 registrations *vs* 55 registrations collected in 2014), which can be interpreted as a signal that we succeeded in reaching a wider audience of researchers interested in participating in the campaign. This result could be also be positively affected by the novelties introduced this year to improve the dissemination of information on EVALITA, e.g. the use of social media such as Twitter and Facebook. Also the number of teams that actually submitted their runs increased in 2016 (34 teams *vs* 23 teams participating in the 2014 edition), even if we reported a substantial gap between the number of actual participants and those who registered.

Table 1: Registered and actual participants

task	**registered**	**actual**
ARTIPHON	6	1
FactA	13	0
NEEL-IT	16	5
QA4FAQ	13	3
PoSTWITA	18	9
SENTIPOLC	24	13
IBM Challenge	6	3
total	96	34

In order to better investigate this issue and gather some insights on the reasons of the significant drop in the number of participants w.r.t. the registrations collected, we ran an online questionnaire specifically designed for those who did not submit any run to the task to which they were registered. In two weeks we collected 14 responses which show that the main obstacles to the actual participation in a task were related to personal issues ("I had an unexpected personal or professional problem outside EVALITA" or

[5] `https://console.ng.bluemix.net/catalog/`

[6] Brazil, France, Germany, India, Ireland, Italy, Mexico, The Netherlands, Spain, Switzerland.

"I underestimated the effort needed") or personal choices ("I gave priority to other EVALITA tasks"). As for this last point, NEEL-it and SENTIPOLC were preferred to FactA, which did not have any participant. Another problem mentioned by some of the respondents is that the evaluation period was too short: this issue is highlighted mostly by those who registered to more than one task.

4 Making Cross-task Shared Data

As an innovation at this year's edition, we aimed at creating datasets that would be shared across tasks so as to provide the community with multi-layered annotated data to test end-to-end systems. In this sense, we encouraged task organisers to annotate the same instances, each task with their respective layer. The involved tasks were: SENTIPOLC, PoSTWITA, NEEL-it and FactA.

The testsets for all four tasks comprise exactly the same 301 tweets, although Sentipolc has a larger testset of 2000 tweets, and FactA has an additional non-social media testset of 597 newswire sentences. Moreover, the training sets of PoSTWITA and NEEL-it are almost entirely subsets of SENTIPOLC. 989 tweets from the 1000 that make NEEL-it's training set are in SENTIPOLC, and 6412 of PoSTWITA (out of 6419) also are included in the SENTIPOLC training set.

Table 2: Overview of cross-task shared data. Number of tweets are reported. When the figure is marked with a *, it is instead the number of sentences from newswire documents.

TRAIN

	SENTIPOLC	NEEL-it	PoSTWITA	FactA
SENTIPOLC	7410	989	6412	0
NEEL-it	989	1000	0	0
PoSTWITA	6412	0	6419	0
FactA	0	0	0	2723*

TEST

	SENTIPOLC	NEEL-it	PoSTWITA	FactA
SENTIPOLC	2000	301	301	301
NEEL-it	301	301	301	301
PoSTWITA	301	301	301	301
FactA	301	301	301	597*+301

The matrix in Table 2 shows both the total number of test instances per task (diagonally) as well as the number of overlapping instances for each task pair. Please note that while SENTIPOLC, NEEL-it, and PoSTWITA provided training and test sets made up entirely of tweets, FactA included tweets only in one of their test set, as a pilot task. FactA's training and standard test sets are composed of newswire data, which we report in terms of number of sentences (Minard et al., 2016). For this reason the number of instances in Table 2 is broken down for FactA's test set: 597 newswire sentences and 301 tweets, the latter being the same as the other tasks.

5 Towards Future Editions

On the basis of this edition's experience, we would like to conclude with a couple of observations that prospective organisers might find useful when designing future editions.

Many novelties introduced in EVALITA 2016 proved to be fruitful in terms of cooperation between academic institutions and industrial companies, balance between research and applications, quantity and quality of annotated data provided to the community. In particular, the involvement of representatives from companies in the organisation of tasks, the development of shared data, the presence of application-oriented tasks and challenge are all elements that could be easily proposed also in future EVALITA editions.

Other innovations can be envisaged for the next campaign. For example, in order to help those who want to participate in more than one task, different evaluation windows for different tasks could be planned instead of having the same evaluation deadlines for all. Such kind of flexibility could foster the participation of teams to multiple tasks, but the fact that it impacts on the work load of the EVALITA's organizers should not be underestimated. Moreover, social media texts turned out to be a very attractive

domain but others could be explored as well. For instance, Humanities resulted as one of the most appealing domains in the questionnaires for industrial companies and former participants and other countries are organising evaluation exercises on it (see, for example, the *Translating Historical Text* shared task at CLIN 27[7]).

References

Leonardo Badino. 2016. The ArtiPhon Challenge at Evalita 2016. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Pierpaolo Basile, Annalina Caputo, Anna Lisa Gentile, and Giuseppe Rizzo. 2016. Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Cristina Bosco, Fabio Tamburini, Andrea Bolioli, and Alessandro Mazzei. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Annalina Caputo, Marco de Gemmis, Pasquale Lops, Franco Lovecchio, and Vito Manzari. 2016. Overview of the EVALITA 2016 Question Answering for Frequently Asked Questions (QA4FAQ) Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Anne-Lyse Minard, Manuela Speranza, and Tommaso Caselli. 2016. The EVALITA 2016 Event Factuality Annotation Task (FactA). In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Rachele Sprugnoli, Viviana Patti, and Franco Cutugno. 2016. Raising Interest and Collecting Suggestions on the EVALITA Evaluation Campaign. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

[7] http://www.ccl.kuleuven.be/CLIN27/

Raising Interest and Collecting Suggestions on the EVALITA Evaluation Campaign

Rachele Sprugnoli
FBK and University of Trento
Via Sommarive, 38123 Trento, Italy
sprugnoli@fbk.eu

Viviana Patti
University of Turin
c.so Svizzera 185, I-10149 Torino, Italy
patti@di.unito.it

Franco Cutugno
University of Naples Federico II
Via Claudio 21, 80126 Naples, Italy
cutugno@unina.it

Abstract

This paper describes the design and reports the results of two questionnaires. The first of these questionnaires was created to collect information about the interest of industrial companies in the field of Italian text/speech analytics towards the evaluation campaign EVALITA; the second to gather comments and suggestions for the future of the evaluation and of its final workshop from the participants and the organizers of the campaign on the last two editions (2011 and 2014). Novelties introduced in the organization of EVALITA 2016 on the basis of the questionnaires results are also reported.

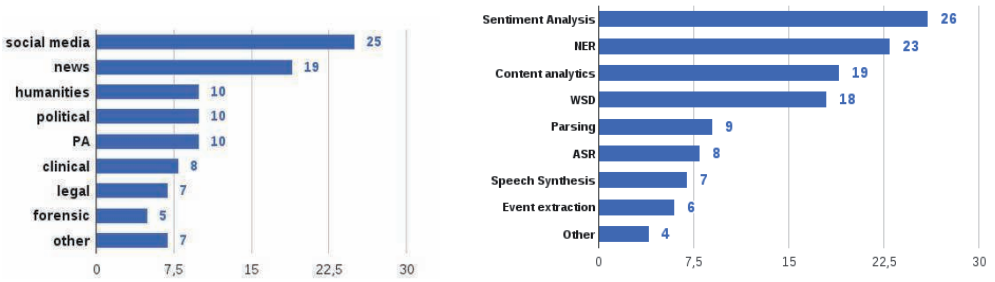
1 Introduction

EVALITA is a periodic evaluation campaign of Natural Language Processing (NLP) and speech tools for the Italian language that has been organized around a set of shared tasks since 2007[1]. Examples of tasks organized in the past EVALITA campaigns are: Named Entities Recognition (NER), Automatic Speech Recognition (ASR), and Sentiment Analysis (Attardi et al., 2015). At the end of the evaluation, a final workshop is organized so to disseminate the results providing participants and organizers with the opportunity to discuss emerging and traditional issues in NLP and Speech technologies for Italian. Over four editions (i.e. 2007, 2009, 2011 and 2014), EVALITA organized more than 30 tasks receiving almost 150 submissions from 90 different organizations: among them 31 (34.4%) were not located in Italy and 10 (11.1%) were not academic. This latter number highlights the limited contribution of enterprises in the campaign, especially in its 2014 edition in which no industrial company was involved as participant. Starting from this observation, in 2015 we designed an online questionnaire to collect information about the interest of industrial companies in the field of text/speech analytics towards EVALITA, with the main aim of understanding how the involvement of companies in the campaign can be fostered.

After four editions we also thought it was time to gather the views of all those who have contributed, until that moment, to the success of EVALITA in order to continuously improve the campaign confirming it as a reference point of the entire NLP and Speech community working on Italian. To this end we prepared another questionnaire for participants and organizers of past EVALITA evaluations to collect comments on the last two editions and receive suggestions for the future of the campaign and of its final workshop.

Questionnaires have been used for different purposes in the NLP community. For example, to carry out user requirements studies before planning long-term investments in the field (Allen and Choukri, 2000; Group, 2010) or in view of the development of a linguistic resource (Oostdijk and Boves, 2006). Moreover, online questionnaires have been adopted to discover trends in the use of a specific technique, e.g. active learning (Tomanek and Olsson, 2009). Similarly to what we propose in this paper, Gonzalo et al. (2002) designed two questionnaires, one for technology developers and one for technology deployers, to acquire suggestions about how to organize Cross-Language Evaluation Forum (CLEF) tasks. As for the feedback received from private companies, the authors report "not very satisfactory results". On the contrary we registered a good number of responses from enterprises in Italy and abroad.

[1] http://www.evalita.it



(a) (b)

Figure 1: (a) Domains and (b) tasks of interest for the companies that answered the questionnaire.

2 Questionnaire for Industrial Companies

The EUROMAP analysis, dated back to 2003, detected structural limits in the Italian situation regarding the Human Language Technology (HLT) market (Joscelyne and Lockwood, 2003). Within that study, 26 Italian HLT suppliers are listed: in 2015, at the time of questionnaire development, only 13 of them were still active. The dynamism of the HLT market in Italy is confirmed by a more recent survey where the activities of 35 Italian enterprises are reported (Di Carlo and Paoloni, 2009): only 18 were still operative in 2015.

Starting from the active private companies present in the aforementioned surveys, we created a repository of enterprises working on Italian text and speech technologies in Italy and abroad. In order to find new enterprises not listed in previous surveys, we took advantage of online repositories (e.g. AngelList[2] and CrunchBase[3]) and of extensive searches on the Web. Our final list included 115 companies among which 57 are not based in Italy. This high number of enterprises dealing with Italian also outside the national borders, reinforces one of the findings of the 2014 Alta Plana survey (Grimes, 2014)[4] that provides a detailed analysis of text analytics market thanks to the answers given to a questionnaire dedicated to technology and solution providers. No Italian company took part in that investigation but Italian resulted as the fourth most analyzed language other than English (after Spanish, German and French) and registered an estimated growth of +11% in two years.

All the companies in our repository were directly contacted via email and asked to fill in the online questionnaire. After an introductory description and the privacy statement, the questionnaire was divided into three sections and included 18 questions. In the first section, we collected information about the company such as its size and nationality; the second had the aim of assessing the interest towards evaluation campaigns in general and towards a possible future participation in EVALITA. Finally, in the third section we collected suggestions for the next edition of EVALITA.

We collected responses from 39 private companies (response rate of 33.9%)[5]: 25 based in Italy (especially in north and central regions) and the rest in other 9 countries[6]. 27 companies work on text technologies, 2 on speech technologies and the remaining declares to do business in both sectors. The great majority of companies (84.6%) has less than 50 employees and, more specifically, 43.6% of them are start-up.

Around 80% of respondents thinks that initiatives for the evaluation of NLP and speech tools are useful for companies and expresses the interest in participating in EVALITA in the future. Motivations behind the negative responses to this last point are related to the fact that the participation to a campaign is considered very time-consuming and also a reputation risk in case of bad results. In addition, EVALITA is perceived as too academically oriented, too focused on general (i.e. non application-oriented) tasks

[2] https://angel.co/

[3] https://www.crunchbase.com/

[4] http://altaplana.com/TA2014

[5] This response rate is in line with the rates reported in the literature on surveys distributed through emails, see (Kaplowitz et al., 2004; Baruch and Holtom, 2008) among others, and with the ones reported in the papers cited in Section 1.

[6] Belgium, Finland, France, Netherlands, Russia, Spain, USA, Sweden, and Switzerland.

and with a limited impact on media. This last problem seems to be confirmed by the percentage of respondents who were not aware of the existence of EVALITA before starting the questionnaire, i.e. 38.5% with 24.1% among Italian companies.

For each of the questions regarding the suggestions for the next campaign (third section), we provided a list of pre-defined options, so to speed up the questionnaire completion, together with a open field for optional additional feedback. Participants could select more than one option. First of all we asked what would encourage and what would prevent the company from participating in the next EVALITA campaign. The possibility of using training and test data also for commercial purposes and the presence of tasks related to the domains of interest for the company have been the most voted options followed by the possibility of advertising for the company during the final workshop (for example by means of exhibition stands or round tables) and the anonymisation of the results so avoiding negative effects on the company image. On the contrary, the lack of time and/or funds is seen as the major obstacle.

Favorite domains and tasks for companies participating in the questionnaire are shown in Figure 1. Social media and news resulted to be the most popular among the domains of interest, followed by humanities, politics and public administration. Domains included in the "Other" category are survey analysis, financial but also public transport and information technology. For what concerns the tasks of interest, sentiment analysis and named entity recognition were the top voted tasks, but a significant interest has been expressed also about content analytics and Word Sense Disambiguation (WSD). In the "Other" category, respondents suggested new tasks such as dialogue analysis, social-network analysis, speaker verification and text classification.

3 Questionnaire for EVALITA Participants and Organizers

The questionnaire for participants and organizers of past EVALITA campaigns was divided into 3 parts. In the first part respondents were required to provide general information such as job position and type of affiliation. In the second part we collected comments about the tasks of past editions asking to rate the level of satisfaction related to four dimensions: (i) the clarity of the guidelines,; (ii) the amount of training data; (iii) the data format; and (iv) the adopted evaluation methods. An open field was also available to add supplementary feedback. Finally, the third section aimed at gathering suggestions for the future of EVALITA posing questions on different aspects, e.g. application domains, type of tasks, structure of the final workshop, evaluation methodologies, dissemination of the results.

The link to the questionnaire was sent to 90 persons who participated in or organized a task in at least one of the last two EVALITA editions. After two weeks we received 39 answers (43.3% response rate) from researchers, Phd candidates and technologists belonging to universities (61.54%) but also to public (25.64%) and private (12.82%) research institutes. No answer from former participants affiliated to private companies was received.

Fifteen out of seventeen tasks of the past have been commented. All the four dimensions taken into consideration obtained positive rates of satisfaction: in particular, 81% of respondents declared to be very o somewhat satisfied by the guidelines and 76% by the format of distributed data. A small percentage of unsatisfied responses (about 13%) were registered on the quantity of training data and on the evaluation. In the open field, the most recurring concern was about the low number of participants in some tasks, sometimes just one or two, especially in the speech ones.

Respondents expressed the will to see some of the old tasks proposed again in the next EVALITA campaign: sentiment polarity classification (Basile et al., 2014), parsing (Bosco et al., 2014), frame labeling (Basili et al., 2013), emotion recognition in speech (Origlia and Galatà, 2014), temporal information processing (Caselli et al., 2014), and speaker identity verification (Aversano et al., 2009). As for the domains of interest, the choices made by participants and organizers are in line with the ones made by industrial companies showing a clear preference for social media (27), news (15), and humanities (13).

The diverging stacked bar chart (Heiberger and Robbins, 2014) in Figure 2, shows how the respondents ranked their level of agreement with a set of statements related to the organization of the final workshop, the performed evaluation and the campaign in general. The majority of respondents agree with almost all statements: in particular, there is a strong consensus about having a demo session during the work-

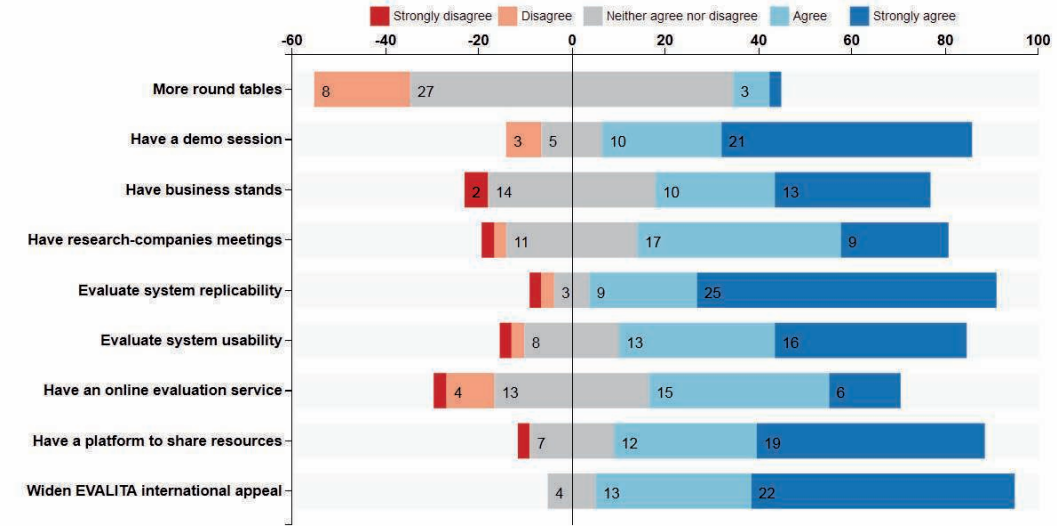


Figure 2: Questionnaire for participants and organizers. Statements assessed on a Likert scale: respondents who agree are on the right side, the ones who disagree on the left and neutral answers are split down the middle. Both percentages and counts are displayed.

shop and also about taking into consideration, during the evaluation, not only systems' effectiveness but also their replicability. Providing the community with a web-based platform to share publicly available resources and systems seems to be another important need as well as enhancing the international visibility of EVALITA. A more neutral, or even negative, feedback was given regarding the possibility of organizing more round tables during the workshop.

4 Lessons Learnt and Impact on EVALITA 2016

Both questionnaires provided us with useful information for the future of EVALITA: they allowed us to acquire input on different aspects of the campaign and also to raise interest towards the initiative engaging two different sectors, the research community and the enterprise community.

Thanks to the questionnaire for industrial companies, we had the possibility to reach and get in touch with a segment of potential participants who weren't aware about the existence of EVALITA or had little knowledge about it. Some of the suggestions coming from enterprises are actually feasible, for example by proposing more application-oriented tasks and by covering domains that are important for them. As for this last point, it is worth noting that the preferred domains are the same for both enterprises and former participants and organizers: this facilitate the design of future tasks based on the collected suggestions. Another issue emerged from both questionnaires is the need of improving the dissemination of EVALITA results in Italy and abroad, in particular outside the boarders of the research community.

The questionnaire for former participants and organizers gave us insights also on practical aspects related to the organization of the final workshop and ideas on how to change the systems evaluation approach taking into consideration different aspects such as replicability and usability.

The results of the questionnaires were presented and discussed during the panel "Raising Interest and Collecting Suggestions on the EVALITA Evaluation Campaign"[7] organized in the context of the second Italian Computational Linguistics Conference[8] (CLiC-it 2015). The panel has sparked an interesting debate on the participation of industrial companies to the campaign, which led to the decision of exploring new avenues for involving industrial stakeholders in EVALITA, as the possibility to call for tasks of industrial interest, that are proposed, and financially supported by the proponent companies. At the same time, the need for a greater internationalization of the campaign, looking for tasks linked to the ones proposed outside Italy, was highlighted. Panelists also wished for an effort in future campaigns towards

[7] http://www.evalita.it/towards2016
[8] https://clic2015.fbk.eu/

the development of shared datasets. Being the manual annotation of data a cost-consuming activity, the monetary contribution of the Italian Association for Computational Linguistics[9] (AILC) was solicited.

The chairs of EVALITA 2016[10] introduced in the organization of the new campaign novel elements, aimed at addressing most of the issues raised by both the questionnaires and the panel (Basile et al., 2016b).

EVALITA 2016 has an application-oriented task (i.e., QA4FAQ) in which representatives of three companies[11] are involved as organizers (Caputo et al., 2016). Another industrial company[12] is part of the organization of another task, i.e., PoSTWITA (Tamburini et al., 2016). Moreover, IBM Italy runs, for the first time in the history of the campaign, a challenge for the development of an app providing monetary awards for the best submissions: the evaluation follows various criteria, not only systems' effectiveness but also other aspects such as intuitiveness and creativity[13]. Given the widespread interest in social media, a particular effort has been put in providing tasks dealing with texts in that domain. Three tasks focus on the processing of tweets (i.e., NEEL-it, PoSTWITA, and SENTIPOLC) and part of the test set is shared among 4 different tasks (i.e., FacTA, NEEL-it, PoSTWITA, and SENTIPOLC) (Minard et al., 2016; Basile et al., 2016a; Barbieri et al., 2016). Part of the SENTIPOLC data was annotated via Crowdflower[14] thanks to funds allocated by AILC.

For what concerns the internationalization issue, in the 2016 edition we had two EVALITA tasks having an explicit link to other shared tasks proposed for English in the context of other evaluation campaigns: the re-run of SENTIPOLC, with an explicit link to the *Sentiment analysis in Twitter* task at SEMEVAL[15], and the new NEEL-it, which is linked to the *Named Entity rEcognition and Linking (NEEL) Challenge* proposed for English tweets at the 6th Making Sense of Microposts Workshop (#Microposts2016, co-located with WWW 2016)[16]. Both tasks have been proposed with the aim to establish a reference evaluation framework in the context of Italian tweets.

We also used social media such as Twitter and Facebook, in order to improve dissemination of information on EVALITA, with the twofold aim to reach a wider audience and to ensure timely communication about various stages of the evaluation campaign.

As for the organization of the final workshop, a demo session is scheduled for the systems participating to the IBM challenge, as a first try to address the request from the community to have new participatory modalities of interacting with systems and teams during the workshop.

Acknowledgments

We are thankful to the panelists and to the audience of the panel 'Raising Interest and Collecting Suggestions on the EVALITA Evaluation campaign' at CLiC-it 2015, for the inspiring and passionate debate. We are also very grateful to Malvina Nissim and Pierpaolo Basile, who accepted with us the challenge to rethink EVALITA and to co-organize the edition 2016 of the evaluation campaign.

References

Jeffrey Allen and Khalid Choukri. 2000. Survey of language engineering needs: a language resources perspective. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC 2000)*.

Giuseppe Attardi, Valerio Basile, Cristina Bosco, Tommaso Caselli, Felice Dell'Orletta, Simonetta Montemagni, Viviana Patti, Maria Simi, and Rachele Sprugnoli. 2015. State of the Art Language Technologies for Italian: The EVALITA 2014 Perspective. *Intelligenza Artificiale*, 9(1):43–61.

[9] http://www.ai-lc.it/

[10] They include Malvina Nissim and Pierpaolo Basile, in addition to the authors of this paper.

[11] QuestionCube:http://www.questioncube.com; AQP:www.aqp.it; SudSistemi: http://www.sudsistemi.eu

[12] CELI: https://www.celi.it/

[13] http://www.evalita.it/2016/tasks/ibm-challenge

[14] https://www.crowdflower.com/

[15] http://alt.qcri.org/semeval2016/task4/

[16] http://microposts2016.seas.upenn.edu/challenge.html

Guido Aversano, Niko Brümmer, and Mauro Falcone. 2009. EVALITA 2009 Speaker Identity Verification Application Track - Organizer's Report. *Proceedings of EVALITA, Reggio Emilia, Italy*.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Yehuda Baruch and Brooks C Holtom. 2008. Survey response rate levels and trends in organizational research. *Human Relations*, 61(8):1139–1160.

Valerio Basile, Andrea Bolioli, Malvina Nissim, Viviana Patti, and Paolo Rosso. 2014. Overview of the Evalita 2014 sentiment polarity classification task. In *Proceedings of the 4th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'14). Pisa, Italy*, pages 50–57. Pisa University Press.

Pierpaolo Basile, Annalina Caputo, Anna Lisa Gentile, and Giuseppe Rizzo. 2016a. Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli. 2016b. EVALITA 2016: Overview of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. aAcademia University Press.

Roberto Basili, Diego De Cao, Alessandro Lenci, Alessandro Moschitti, and Giulia Venturi. 2013. Evalita 2011: the frame labeling over Italian texts task. In *Evaluation of Natural Language and Speech Tools for Italian*, pages 195–204. Springer.

Cristina Bosco, Felice Dell'Orletta, Simonetta Montemagni, Manuela Sanguinetti, and Maria Simi. 2014. The EVALITA 2014 dependency parsing task. *Proceedings of EVALITA*.

Annalina Caputo, Marco de Gemmis, Pasquale Lops, Franco Lovecchio, and Vito Manzari. 2016. Overview of the EVALITA 2016 Question Answering for Frequently Asked Questions (QA4FAQ) Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Tommaso Caselli, Rachele Sprugnoli, Manuela Speranza, and Monica Monachini. 2014. EVENTI: EValuation of Events and Temporal INformation at Evalita 2014. In *Proceedings of the First Italian Conference on Computational Linguistics CLiC-it 2014 & and of the Fourth International Workshop EVALITA 2014*, pages 27–34. Pisa University Press.

Andrea Di Carlo and Andrea Paoloni. 2009. *Libro Bianco sul Trattamento Automatico della Lingua.*, volume 1. Fondazione Ugo Bordoni, Roma.

Julio Gonzalo, Felisa Verdejo, Anselmo Peñas, Carol Peters, Khalid Choukri, and Michael Kluck. 2002. Cross Language Evaluation Forum - User Needs: Deliverable 1.1.1. Technical report.

Seth Grimes. 2014. Text analytics 2014: User perspectives on solutions and providers. *Alta Plana.*

FLaReNet Working Group. 2010. Results of the questionnaire on the priorities in the field of language resources. Technical report, Department of Computer Science, Michigan State University, September.

Richard M Heiberger and Naomi B Robbins. 2014. Design of diverging stacked bar charts for likert scales and other applications. *Journal of Statistical Software*, 57(5):1–32.

Andrew Joscelyne and Rose Lockwood. 2003. *Benchmarking HLT progress in Europe.*, volume 1. The EUROMAP Study, Copenhagen.

Michael D Kaplowitz, Timothy D Hadlock, and Ralph Levine. 2004. A comparison of web and mail survey response rates. *Public opinion quarterly*, 68(1):94–101.

Anne-Lyse Minard, Manuela Speranza, and Tommaso Caselli. 2016. The EVALITA 2016 Event Factuality Annotation Task (FactA). In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Nelleke Oostdijk and Lou Boves. 2006. User requirements analysis for the design of a reference corpus of written dutch. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation(LREC 2006)*.

Antonio Origlia and Vincenzo Galatà. 2014. EVALITA 2014: Emotion Recognition Task (ERT). In *Proceedings of the First Italian Conference on Computational Linguistics CLiC-it 2014 & and of the Fourth International Workshop EVALITA 2014*, pages 112–115. Pisa University Press.

Fabio Tamburini, Cristina Bosco, Alessandro Mazzei, and Andrea Bolioli. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Katrin Tomanek and Fredrik Olsson. 2009. A web survey on the use of active learning to support annotation of text data. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 45–48. Association for Computational Linguistics.

Keynote: Profiling the Personality of Social Media Users

Walter Daelemans
CLiPS, University of Antwerp, The Netherlands
walter.daelemans@uantwerpen.be

In the social media, everybody is a writer, and many people freely give away their personal information (age, gender, location, education, and, often indirectly, also information about their psychology such as personality, emotions, depression etc.). By linking the text they write with this metadata of many social media users, we have access to large amounts of rich data about real language use. This makes possible the development of new applications based on machine learning, as well as a new empirical type of sociolinguistics based on big data.

In this paper I will provide a perspective on the state of the art in profiling social media users focusing on methods for personality assignment from text. Despite some successes, it is still uncertain whether this is even possible, but if it is, it will allow far-reaching applications. Personality is an important factor in life satisfaction and determines how we act, think and feel. Potential applications include targeted advertising, adaptive interfaces and robots, psychological diagnosis and forensics, human resource management, and research in literary science and social psychology.

I will describe the personality typology systems currently in use (MBTI, Big Five, Enneagram), the features and methods proposed for assigning personality, and the current state of the art, as witnessed from, for example, the PAN 2015 competition on profiling and other shared tasks on benchmark corpora. I will also go into the many problems in this subfield of profiling; for example the unreliability of the gold standard data, the shaky scientific basis of the personality typologies proposed, and the low accuracies achieved for many traits in many corpora. In addition, as is the case for the larger field of profiling, we are lacking sufficiently large balanced corpora for studying the interaction with topic and register, and the interactions between profile dimensions such as age and gender with personality.

As a first step toward a multilingual shared task on personality profiling, I will describe joint work with Ben Verhoeven and Barbara Plank on collecting and annotating the TwiSty corpus (Verhoeven et al., 2016). TwiSty (http://www.clips.ua.ac.be/datasets/twisty-corpus) contains personality (MBTI) and gender annotations for a total of 18,168 authors spanning six languages: Spanish, Portuguese, French, Dutch, Italian, German. A similar corpus also exists for English. It may be a first step in the direction of a balanced, multilingual, rich social media corpus for profiling.

References

Ben Verhoeven, Walter Daelemans, and Barbara Plank. 2016. Twisty: A multilingual twitter stylometry corpus for gender and personality profiling. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association (ELRA).

Bio

Walter Daelemans is professor of Computational Linguistics at the University of Antwerp where he directs the CLiPS computational linguistics research group. His research interests are in machine learning of natural language, computational psycholinguistics, computational stylometry, and language technology applications, especially biomedical information extraction and cybersecurity systems for social networks. He has supervised 25 finished PhDs and (co-)authored more than 300 publications. He was elected EURAI Fellow, ACL Fellow, and member of the Royal Academy for Dutch Language and Literature.

PART II

EVALITA 2016: TASK OVERVIEWS AND PARTICIPANT REPORTS

The ArtiPhon Task at Evalita 2016

Leonardo Badino
Center for Translational Neurophysiology of Speech and Communication
Istituto Italiano di Tecnologia – Italy
leonardo.badino@iit.it

Abstract

English. Despite the impressive results achieved by ASR technology in the last few years, state-of-the-art ASR systems can still perform poorly when training and testing conditions are different (e.g., different acoustic environments). This is usually referred to as the mismatch problem. In the ArtiPhon task at Evalita 2016 we wanted to evaluate phone recognition systems in mismatched speaking styles. While training data consisted of read speech, most of testing data consisted of single-speaker hypo- and hyper-articulated speech. A second goal of the task was to investigate whether the use of speech production knowledge, in the form of measured articulatory movements, could help in building ASR systems that are more robust to the effects of the mismatch problem. Here I report the result of the only entry of the task and of baseline systems.

Italiano. *Nonostante i notevoli risultati ottenuti recentemente nel riconoscimento automatico del parlato (ASR) le prestazioni dei sistemi ASR peggiorano significativamente in quando le condizioni di testing sono differenti da quelle di training (per esempio quando il tipo di rumore acustico è differente). Un primo gol della ArtiPhon task ad Evalita 2016 è quello di valutare il comportamento di sistemi di riconoscimento fonetico in presenza di un mismatch in termini di registro del parlato. Mentre il parlato di training consiste di frasi lette ad un velocita; di eloquio "standard", il parlato di testing consiste di frasi sia iperche ipo-articolate. Un secondo gol della task è quello di analizzare se e come l'utilizzo di informazione concernente la produzione del parlato migliora l'accuratezza dell'ASR e in particolare nel caso di mismatch a livello di registri del parlato. Qui riporto risultati dell'unico sistema che è stato sottomesso e di una baseline.*

1 Introduction

In the last five years ASR technology has achieved remarkable results, thanks to increased training data, computational resources, and the use of deep neural networks (DNNs, (LeCun et al., 2015)). However, the performance of connectionist ASR degrades when testing conditions are different from training conditions (e.g., acoustic environments are different) (Huang et al., 2014). This is usually referred to as the training-testing mismatch problem. This problem is partly masked by multi-condition training (Seltzer et al., 2013) that consists in using very large training datasets (up to thousands of hours) of transcribed speech to cover as many sources of variability as possible (e.g., speaker's gender, age and accent, different acoustic environments).

One of the two main goals of the ArtiPhon task at Evalita 2016 was to evaluate phone recognition systems in mismatched speaking styles. Between training and testing data the speaking style was the condition that differed. More specifically, while the training dataset consists of read speech where the speaker was required to keep a constant speech rate, testing data range from slow and hyper-articulated speech to fast and hypo-articulated speech. Training and testing data are from the same speaker.

The second goal of the ArtiPhon task was to investigate whether the use of speech production knowledge, in the form of measured articulatory movements, could help in building ASR systems that are more robust to the effects of the mismatch problem.

The use of speech production knowledge, i.e., knowledge about how the vocal tract behaves when it produces speech sounds, is motivated by the fact that complex phenomena

observed in speech, for which a simple purely acoustic description has still to be found, can be easily and compactly described in speech production-based representations. For example, in Articulatory Phonology (Browman and Goldstein, 1992) or in the distinctive features framework (Jakobson et al., 1952) coarticulation effects can be compactly modeled as temporal overlaps of few vocal tract gestures. The vocal tract gestures are regarded as invariant, i.e., context- and speaker-independent, production targets that contribute to the realization of a phonetic segment. Obviously the invariance of a vocal tract gesture partly depends on the degree of abstraction of the representation but speech production representations offer compact descriptions of complex phenomena and of phonetic targets that purely acoustic representations are not able to provide yet (Maddieson, 1997).

Recently, my colleagues and I have proposed DNN-based "articulatory" ASR where the DNN that computes phone probabilities is forced, during training, to learn/use motor features. We have proposed strategies that allow motor information to produce an inductive bias on learning. The bias resulted in improvements over strong DNN-based purely auditory baselines, in both speaker-dependent (Badino et al., 2016) and speaker-independent settings (Badino, 2016)

Regarding the Artiphon task, unfortunately only one out of the 6 research groups that expressed an interest in the task actually participated (Piero Cosi from ISTC at CNR, henceforth I will refer to this participant as ISTC) (Cosi, 2016). The ISTC system did not use articulatory data.

In this report I will present results of the ISTC phone recognition systems and of baseline systems that also used articulatory data.

2 Data

The training and testing datasets used for the ArtiPhon task were selected from voice cnz of the Italian MSPKA corpus (http://www.mspkacorpus.it/) (Canevari et al., 2015), which was collected in 2015 at the Istituto Italiano di Tecnologia (IIT).

The training dataset corresponds to the 666-utterance session 1 of MPSKA, where the speaker was required to keep a constant speech rate. The testing dataset was a 40-utterance subset selected from session 2 of MPSKA. Session 2 of MPSKA contains a continuum of ten descending articulation degrees, from hyper-articulated to hypo-articulated speech. Details on the procedure used to elicit this continuum are provided in (Canevari et al., 2015).

Articulatory data consist of trajectories of 7 vocal tract articulators and recorded with the NDI (Northern Digital Instruments, Canada) wave speech electromagnetic articulography system at 400 Hz.

Seven 5-Degree-of-freedom (DOF) sensor coils were attached to upper and lower lips (UL and LL), upper and lower incisors (UI and LI), tongue tip (TT), tongue blade (TB) and tongue dorsum (TD). For head movement correction a 6-DOF sensor coil was fixed on the bridge of a pair of glasses worn by the speakers.

The NDI system tracks sensor coils in 3D space providing 7 measurements per each coil: 3 positions (i.e. x; y; z) and 4 rotations (i.e. Q0;Q1;Q2;Q3) in quaternion format with Q0 = 0 for 5-DOF sensor coils.

Contrarily to other articulographic systems (e.g. Carstens 2D AG200, AG100) speakers head is free to move. That increases comfort and the naturalness of speech.

During recordings speakers were asked to read aloud each sentence that is prompted on a computer screen. In order to minimize disfluencies speakers had time to silently read each sentence before reading out.

The audio files of the MSPKA corpus are partly saturated.

The phone set consists of 60 phonemes, although the participants could collapsed them to 48 phonemes as proposed in (Canevari et al., 2015).

3 Sub-tasks

In Artiphon sub-tasks are phone recognition tasks. The participants were asked to:

- train phone recognition systems on the training dataset and then run them on the test dataset;
- (optional) use articulatory data to build "articulatory" phone recognition systems.

Articulatory data were also provided in the test dataset thus three different scenarios were possible:

- Scenario 1. Articulatory data not available
- Scenario 2. Articulatory data available at both training and testing.

- Scenario 3. Articulatory data available only at training.

Note that only scenarios 1 and 3 are realistic ASR scenarios as during testing articulatory data are very difficult to access.

Participants could build purely acoustic and articulatory phone recognition systems starting from the Matlab toolbox developed at IIT, available at https://github.com/robotology/natural-speech.

4 Phone recognition systems

Baseline systems are hybrid DNN-HMM systems while ISTC systems are either GMM-HMM or DNN-HMM systems with DNN-HMM.

The ISTC systems were trained using the KALDI ASR engine. ISTC systems used either the full phone set (with 60 phone labels) or a reduced phone set (with 29 phones). In the reduced phone set all phones that are not actual phonemes in current Italian were correctly removed. However, important phonemes were also arbitrarily removed, most importantly, geminates and corresponding non-geminate phones were collapsed into a single phone (e.g., /pp/ and /p/ were both represented by label /p/).

ISCT systems used either monophones or triphones.

ISCT systems were built using KALDI (Povey et al., 2011) with TIMIT recipes adapted to the APASCI dataset (Angelini & al., 1994). Two training datasets where used:

- the single-speaker dataset provided within the ArtiPhon task;
- the APASCI dataset.

In all cases only acoustic data were used (scenario 1), so the recognition systems were purely acoustic recognition systems. Henceforth I will refer to ISTC systems trained on the ArtiPhon single-speaker training dataset as speaker-dependent ISTC systems (as the speaker in training and testing data is the same) and to ISTC systems trained on the APASCI dataset as speaker-independent ISTC systems.
Baseline systems were built using the aforementioned Matlab toolbox and only trained on the ArtiPhon training dataset (so they are all speaker-dependent systems).

Baseline systems used a 48 phone set and three-state monophones (Canevari et al., 2015). Baseline systems were trained and tested according to all three aforementioned three scenarios. The articulatory data considered only refer to x-y positions of 6 coils (the coil attached to the upper teeth was exluded).

5 Results

Here I report some of the most relevant results regarding ISTC and baseline systems.

Baseline systems and ISTC systems are not directly comparable as very different assumptions were made, most importantly they use different phone sets.

Additionally, ISCT systems were mainly concerned with exploring the best performing systems (created using well-known KALDI recipes for ASR) and comparing them in the speaker-dependent and in the speaker-independent case.

Baselines systems were created to investigate the utility of articulatory features in mismatched speaking styles.

5.1 ISCT systems

Here I show results on ISCT systems trained and tested on the 29 phone set. Table 1 shows results of the speaker-dependent systems while Table 2 shows results in the speaker-independent case.

The results shown in the two tables refer to the various training and decoding experiments, see (Rath et al., 2013) for all acronyms references:

- MonoPhone (mono);
- Deltas + Delta-Deltas (tri1);
- LDA + MLLT (tri2);
- LDA + MLLT + SAT (tri3);
- SGMM2 (sgmm2_4);
- MMI + SGMM2 (sgmm2_4_mmi_b0.1-4);
- Dan's Hybrid DNN (tri4-nnet),
- system combination, that is Dan's DNN + SGMM (combine_2_1-4);
- Karel's Hybrid DNN (dnn4_pretrain-dbn_dnn);
- system combination that is Karel's DNN + sMBR (dnn4_pretrain-dbn_dnn_1-6).

	Training & Decoding	%PCR	%SUB	%DEL	%INS	%PER
MonoPhone	mono	80.1	11.0	8.9	2.6	22.4
Delta + Delta-Deltas	tri1	85.4	7.7	6.9	2.6	17.2
LDA + MLTT	tri2	85.8	7.3	6.9	2.4	16.6
LDA + MLTT + SAT (SI)	tri3.si	85.2	7.5	7.3	2.7	17.6
LDA + MLTT + SAT	tri3	86.7	6.5	6.8	2.1	15.3
sgmm2_4: SGMM2	sgmm2_4	87.2	6.5	6.3	2.3	15.1
MMI + SGMM2 (iteration n.1)	sgmm2_4_mmi_b0.1	87.2	6.5	6.3	2.3	15.1
MMI + SGMM2 (iteration n.2)	sgmm2_4_mmi_b0.2	86.8	6.3	6.9	1.9	15.0
MMI + SGMM2 (iteration n.3)	sgmm2_4_mmi_b0.3	87.4	6.3	6.3	2.3	14.9
MMI + SGMM2 (iteration n.4)	sgmm2_4_mmi_b0.4	87.4	6.3	6.3	2.3	14.9
DNN Hybrid (Dan's)	tri4-nnet	82.8	8.5	8.8	2.4	19.7
SGMM + DNN Hybrid (Dan's) (it. 1)	combine_2 (1)	87.4	6.1	6.5	2.5	15.1
SGMM + DNN Hybrid (Dan's) (it. 2)	combine_2 (2)	87.4	6.1	6.5	2.5	15.1
SGMM + DNN Hybrid (Dan's) (it. 3)	combine_2 (3)	87.3	6.1	6.6	2.5	15.2
SGMM + DNN Hybrid (Dan's) (it. 4)	combine_2 (4)	87.3	6.1	6.6	2.5	15.2
DNN Hybrid (Karel's)	dnn4_pretrain-dbn_dnn	86.1	6.8	7.1	2.3	16.2
DNN Hybrid (Karel's), sMBR training (it. 1)	dnn4_pretrain-dbn_dnn_smbr (1)	86.1	6.8	7.2	2.3	16.2
DNN Hybrid (Karel's), sMBR training (it. 6)	dnn4_pretrain-dbn_dnn_smbr (6)	86.0	6.6	7.4	2.2	16.2

Table 1. Results of ISCT systems on speaker-dependent sub-task with the 29-phone set.
MFCC: Mel-Frequency Cepstral Coefficients; LDA: Linear Discriminant Analysis; MLTT: Maximum Likelihood Linear Transform; fMLLR: feature space Maximum Likelihood Linear Regression; CMN: Cepstral Mean Normalization. MMI: Maximum Mutual Information; BMMI: Boosted MMI; MPE: Minimum Phone Error; sMBR: State-level Minimum Bayes Risk

	Training & Decoding	%PCR	%SUB	%DEL	%INS	%PER
MonoPhone	mono	61,3	23.8	14.9	2.3	41.0
Delta + Delta-Deltas	tri1	66.8	21.3	11.9	3.7	36.9
LDA + MLTT	tri2	70.0	19.5	10.4	4.6	34.5
LDA + MLTT + SAT (SI)	tri3.si	70.2	18.3	11.5	2.2	32.0
LDA + MLTT + SAT	tri3	74.5	16.8	8.7	3.0	28.4
sgmm2_4: SGMM2	sgmm2_4	75.7	15.3	9.0	4.4	28.7
MMI + SGMM2 (iteration n.1)	sgmm2_4_mmi_b0.1	75.7	15.2	9.1	4.1	28.4
MMI + SGMM2 (iteration n.2)	sgmm2_4_mmi_b0.2	76.3	15.6	8.1	4.7	28.4
MMI + SGMM2 (iteration n.3)	sgmm2_4_mmi_b0.3	76.3	15.5	8.2	4.5	28.2
MMI + SGMM2 (iteration n.4)	sgmm2_4_mmi_b0.4	76.3	15.4	8.3	4.6	28.2
DNN Hybrid (Dan's)	tri4-nnet	70.7	17.3	12.0	3.7	31.8
SGMM + DNN Hybrid (Dan's) (it. 1)	combine_2 (1)	76.1	15.2	8.7	3.7	27.5
SGMM + DNN Hybrid (Dan's) (it. 2)	combine_2 (2)	76.2	15.0	8.8	3.6	27.4
SGMM + DNN Hybrid (Dan's) (it. 3)	combine_2 (3)	76.1	15.1	8.8	3.5	27.4
SGMM + DNN Hybrid (Dan's) (it. 4)	combine_2 (4)	76.2	15.2	8.6	3.4	27.1
DNN Hybrid (Karel's)	dnn4_pretrain-dbn_dnn	75.6	14.6	9.8	2.4	26.9
DNN Hybrid (Karel's), sMBR training (it. 1)	dnn4_pretrain-dbn_dnn_smbr (1)	75.3	14.6	10.2	2.3	27.1
DNN Hybrid (Karel's), sMBR training (it. 6)	dnn4_pretrain-dbn_dnn_smbr (6)	75.6	14.7	9.7	2.5	27.0

Table 2. Results of ISCT systems on speaker independent sub-task with 29-phone set.

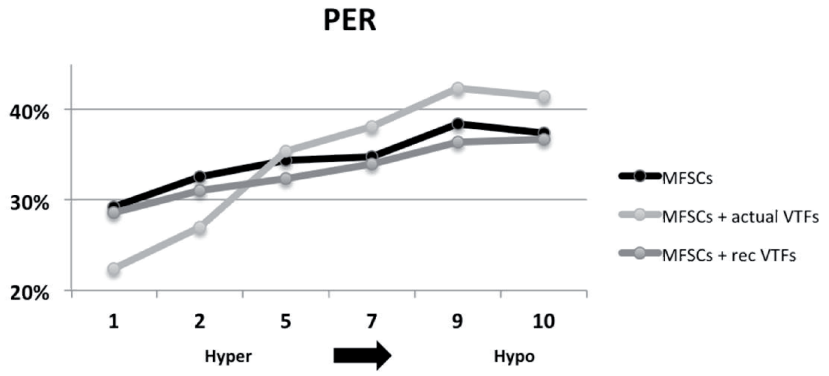


Figure 1. Phone Error Rate (PER) over 10 degrees of articulation when (i) using only MFSCs (black), (ii) using MFSCs appended to actual VTFs (light grey) and (iii) using MFSCs appended to recovered VTFs (dark grey)

The most interesting result is that while DNN-HMM systems outperform GMM-HMM systems in the speaker-independent case (as expected), GMM-HMM and more specifically sub-space GMM-HMM (Povey et al., 2011), outperform the DNN-based systems in the speaker dependent case.

Another interesting result is that sequence based training strategies (Vesely et al., 2013) did not produce any improvement over frame-based training strategies.

5.2 Baseline systems – acoustic vs. articulatory results

The baseline systems addressed the two main questions that motivated the design of the ArtiPhon task: (i) how does articulatory information contribute to phone recognition?; (ii) how does the phone recognition system performance vary along the continuum from hyper-articulated speech to hypo-articulated speech?

Figure 1 shows phone recognition results of 3 different systems over the 10 degrees of articulation from hyper-articulated to hypo-articulated speech.

The three systems, reflecting the three aforementioned training-testing scenarios, are:

- phone recognition system that only uses acoustic feature, specifically mel-filtered spectra coefficients (MFSCs, scenario 1)
- articulatory phone recognition system where actual measured articulatory/vocal tract features (VTFs) are appended to the input acoustic vector during testing (scenario 2)
- articulatory phone recognition system where reconstructed VTFs are appended to the input acoustic vector during testing (scenario 3)

The last system reconstructs the articulatory features using an acoustic-to-articulatory mapping learned during training (see, e.g., (Canevari et al., 2013)for details).

All systems used a 48-phone set as in cc.

One first relevant result is that all systems performed better at high levels of hyper-articulation than at "middle" levels (i.e., levels 5-6) which mostly corresponds to the training condition (Canevari et al., forthcoming). In all systems performance degraded from hyper- to hypo-articulated speech.

Reconstructed VTFs always decrease the phone error rate. Appending recovered VTFs to the acoustic feature vector produces a relative PER reduction that ranges from 4.6% in hyper-articulated speech, to 5.7% and 5.2% in middle- and hypo-articulated speech respectively.

Actual VTFs provide a relative PER reduction up to 23.5% in hyper-articulated speech, whereas, unexpectedly, no improvements are observed when actual VTFs are used in middle- and hypo-articulated speech. That might due to the fact that sessions 1 and 2 of the MSPKA corpus took place in different days so EMA coils could be in slightly different positions.

6 Conclusions

This paper described the ArtiPhon task at Evalita 2016 and showed and discussed results of baseline phone recognition systems and of the submitted systems.

References

Angelini, B., Brugnara, F., Falavigna, D., Giuliani, D., Gretter, R., and Omologo, M.. 1994. Speaker Independent Continuous Speech Recognition Using an Acoustic-Phonetic Italian Corpus. In *Proceedings of ICSLP*. Yokohama, Japan.

Badino, L. 2016. Phonetic Context Embeddings for DNN-HMM Phone Recognition. In *Proceedings of Interspeech*. San Francisco, CA.

Badino, L., Canevari, C., Fadiga, L., and Metta, G. 2016. Integrating Articulatory Data in Deep Neural Network-based Acoustic Modeling. *Computer Speech and Language*, 36, 173–195.

Browman, C., and Goldstein, L. 1992. Articulatory phonology: an overview. *Phonetica* 49 (3–4), 155–180.

Canevari, C., Badino, L., and Fadiga, L. 2015. A new Italian dataset of parallel acoustic and articulatory data. *Proceedings of Interspeech*. Dresden.

Canevari, C., Badino, L., D'Ausilio, A., and Fadiga, L. Forthcoming. Analysis of speech production differences between Hypo-and Hyper-articulated speech and implications for Articulatory ASR. Submitted.

Canevari, C., Badino, L., Fadiga, L., and Metta, G. 2013. Cross-corpus and cross-linguistic evaluation of a speaker-dependent DNN-HMM ASR sytem using EMA data. In *Proceedings of Workshop on Speech Production for Automatic Speech Recognition*. Lyon, France.

Cosi, P. 2016. Phone Recognition Experiments on ArtiPhon with KALDI. In *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. Napoli.

Huang, Y., Yu, D., Liu, C., and Gong, Y. 2014. A Comparative Analytic Study on the Gaussian Mixture and Context Dependent Deep Neural Network Hidden Markov Models. *Proceedings of Interspeech*. Singapore.

Jakobson, R., Fant, G., and Halle, M. 1952. *Preliminaries to Speech Analysis: The Distinctive Features and Their Correlates.* Cambridge,MA: MIT Press.

LeCun, Y., Bengio, Y., and Hinton, G. E. 2015. Deep Learning. *Nature*, 521, 436-444.

Maddieson, I. 1997. Phonetic universals. In W. Hardcastle, and J. Laver, *The Handbook of Phonetic Sciences*. pp. 619--639. Oxford: Blackwell Publishers.

Povey, D., Burget, L., Agarwal, M., Akyazi, P., Kai, F., Ghoshal, A., Glembek, O., Goel, N., Karafiát, M., Rastrow , A., Rosei, R. C., Schwarzb, P., and Thomash, S. 2011. The Subspace Gaussian Mixture Model - a Structured Model for Speech Recognition. *ComputerSpeech and Language*. 25(2), 404-439.

Povey, D., Ghoshal, A., & al. 2011. The KALDI Speech Recognition Toolkit . *Proceedings of ASRU2011*.

Rath, S. P., Povey, D., Vesely, K., and Cernocky, J. 2013. Improved feature processing for Deep Neural Networks. *Proceedings of Interspeech*, pp. 109--113. Lyon, France.

Seltzer, M., Yu, D., and Wang, Y. 2013. An Investigation of Deep Neural Networks for Noise Robust Speech Recognition. *Proceedings of ICASSP*. Vancouver, Canada.

Vesely, K., Ghoshal, A., Burget, L., and Povey, D. 2013. Sequence-discriminative training of deep neural networks. *Proceeding of Interspeech*. Lyon, France.

Phone Recognition Experiments on ArtiPhon with KALDI

Piero Cosi
Istituto di Scienze e Tecnologie della Cognizione
Consiglio Nazionale delle Ricerche
Sede Secondaria di Padova – Italy
piero.cosi@pd.istc.cnr.it

Abstract

English. In this work we present the results obtained so far in different recognition experiments working on the audio only part of the ArtiPhon corpus used for the EVALITA 2016 speech-mismatch ArtiPhone challenge.

Italiano. *In questo lavoro si presentano i risultati ottenuti sinora in diversi esperimenti di riconoscimento fonetico utilizzanti esclusivamente la sola parte audio del corpus ArtiPhon utilizzato per il challenge ArtiPhone di EVALITA 2016.*

1 Introduction

In the last few years, the automatic speech recognition (ASR) technology has achieved remarkable results, mainly thanks to increased training data and computational resources. However, ASR trained on thousand hours of annotated speech can still perform poorly when training and testing conditions are different (e.g., different acoustic environments). This is usually referred to as the mismatch problem.

In the ArtiPhon challenge participants will have to build a speaker-dependent phone recognition system that will be evaluated on mismatched speech rates. While training data consists of read speech where the speaker was required to keep a constant speech rate, testing data range from slow and hyper-articulated speech to fast and hypo-articulated speech.

The training dataset contains simultaneous recordings of audio and vocal tract (i.e., articulatory) movements recorded with an electromagnetic articulograph (Canevari et al., 2015).

Participants were encouraged to use the training articulatory data to increase the generalization performance of their recognition system. However, we decided not to use them, mainly for the sake of time, but also because we wanted to compare the results with those obtained in the past on different adult and children speech audio-only corpora (Cosi & Hosom, 2000; Cosi & Pellom, 2005; Cosi, 2008; Cosi, 2009; Cosi et al., 2014; Cosi et al., 2015).

2 Data

We received the ArtiPhon (Canevari et al., 2015) training data by the Istituto Italiano di Tecnologia - Center for Translational Neurophysiology of Speech and Communication (CTNSC) late in July 2016, while the test material was released at the end of September 2016. The ArtiPhon dataset contains the audio and articulatory data recorded from three different speakers in citation condition. In particular for the EVALITA 2016 ArtiPhon - Articulatory Phone Recognition challenge only one speaker (cnz - 666 utterances) was considered.

The audio was sampled at 22050 Hz while articulatory data were extracted by the use of the NDI (Northen Digital Instruments, Canada) wave speech electromagnetic articulograph at 400 Hz sampling rate.

Four subdirectories are available:

- wav_1.0.0: each file contains an audio recording
- lab_1.0.0: each file contains phonetic labels automatically computed using HTK
- ema_1.0.0: each file contains 21 channels: coordinate in 3D space (xul yul zul xll yll zll xui yui zui xli yli zli xtb ytb ztb xtm ytm ztm xtt ytt ztt)

Head movement correction was automatically performed. First an adaptive median filter with a window from 10 ms to 50 ms and secondly a smooth elliptic low-pass filter with 20 Hz cutoff frequency were applied to each channel.

Unfortunately, we discovered that the audio data was completely saturated both in the training and the test set, thus forcing us to develop various experiments both using the full set of phonemes but also a smaller reduced set in order to make more effective and reliable the various phone recognition experiments.

3 ASR

DNN has proven to be an effective alternative to HMM - Gaussian mixture modelisation (GMM) based ASR (HMM-GMM) (Bourlard and Morgan, 1994; Hinton et al., 2012) obtaining good performance with context dependent hybrid DNN-HMM (Mohamed et al., 2012; Dahl et al., 2012).

Deep Neural Networks (DNNs) are indeed the latest hot topic in speech recognition and new systems such as KALDI (Povey et al., 2011) demonstrated the effectiveness of easily incorporating "Deep Neural Network" (DNN) techniques (Bengio, 2009) in order to improve the recognition performance in almost all recognition tasks.

DNNs has been already applied on different adults and children Italian speech corpora, obtaining quite promising results (Cosi, 2015; Serizel & Giuliani, 2014; Serizel & Giuliani, 2016).

In this work, the KALDI ASR engine adapted to Italian was adopted as the target ASR system to be evaluated on the ArtiPhon data set.

At the end we decided not to use the articulatory data available in the ArtiPhon data set, because we wanted to compare the final results of this challenge with those obtained in the past on different audio-only corpora which were not characterized by the above cited speech mismatch problem.

4 The EVALITA 2016 - ArtiPhon Challenge

A speaker dependent experiment characterized by training and test speech type mismatch was prepared by using the ArtiPhon challenge training and test material. A second speaker independent experiment was also set by testing the ArtiPhon test data using a previously trained ASR acoustic model on APASCI (Angelini et al., 1994), thus having in this case both speech type and speaker mismatch.

For both experiments, we used the KALDI ASR engine, and we started from the TIMIT recipe, which was adapted to the ArtiPhon Italian data set.

Deciding when a phone should be considered incorrectly recognized was another evaluation issue. In this work, as illustrated in Table 1, two set of phones, with 29 and 60 phones respectively, have been selected for the experiments, even if the second set is far from being realistic given the degraded quality of the audio signal.

60	29	60	29	60	29
a	a	j	j	pp	p
a1	a	J	J	r	r
b	b	JJ	J	rr	r
bb	b	k	k	s	s
d	d	kk	k	ss	s
dd	d	l	l	S	S
dz	dz	ll	l	SS	S
ddz	dz	L	L	t	t
dZ	dZ	LL	L	tt	t
ddZ	dZ	m	m	ts	ts
e	e	mm	m	tts	ts
e1	e	ng	n	tS	tS
E	e	nf	n	ttS	tS
E1	e	n	n	u	u
f	f	nn	n	u1	u
ff	f	o	o	v	v
g	g	o1	o	vv	v
gg	g	O	o	w	w
i	i	O1	o	z	z
i1	i	p	p	sil	sil

Table 1: 60 and 29 phones set (SAMPA).

Considering that, in unstressed position, the oppositions /e/ - /E/ and /o/ - /O/ are often neutralized in the Italian language, it was decided to merge these couples of phonemes. Since the occurrences of /E/ and /O/ phonemes were so rare in the test set, this simplification have had no influence in the test results.

Then, the acoustic differences between stressed (a1, e1, E1, i1, o1, O1, u1) and unstressed vowels (a, e, E, i, o, O, u) in Italian are subtle and mostly related to their duration. Furthermore, most of the Italian people pronounce vowels according to their regional influences instead of "correct-standard" pronunciation, if any, and this sort of inaccuracies is quite common. For these reasons, recognition outputs have been evaluated using the full 60-phones ArtiPhon set as well as a more realistic reduced 29-phones set, which do not count the mistakes between stressed and unstressed vowels,

geminates vs single phones and /ng/ and /nf/ all-phones vs the /n/ phoneme.

In Table 2, the results of the EVALITA 2016 ArtiPhon speaker dependent experiment with the 60-phones and 29-phones are summarized in Table 2a and 2b respectively, for all the KALDI ASR engines, as in the TIMIT recipe.

Training & Decoding	%PCR	%SUB	%DEL	%INS	%PER
MonoPhone	mono	61.9	29.9	8.2	3.3	41.3
Delta + Delta-Deltas	tri1	66.4	25.9	7.6	2.4	35.9
LDA + MLTT	tri2	66.1	25.8	8.1	2.5	36.4
LDA + MLTT + SAT (SI)	tri3.si	67.1	25.4	7.5	2.6	35.5
LDA + MLTT + SAT	tri3	67.9	25.5	6.6	1.9	34.0
sgmm2_4: SGMM2	sgmm2_4	68.7	24.5	6.8	1.7	33.1
MMI + SGMM2 (iteration n.1)	sgmm2_4_mmi_b0.1	68.8	24.6	6.6	1.7	32.9
MMI + SGMM2 (iteration n.2)	sgmm2_4_mmi_b0.2	68.7	24.6	6.7	1.6	32.9
MMI + SGMM2 (iteration n.3)	sgmm2_4_mmi_b0.3	68.7	24.5	6.8	1.6	32.9
MMI + SGMM2 (iteration n.4)	sgmm2_4_mmi_b0.4	68.8	24.5	6.8	1.6	32.8
DNN Hybrid (Dan's)	tri4-nnet	64.6	27.7	7.7	3.0	38.3
SGMM + DNN Hybrid (Dan's) (it. 1)	combine_2 (1)	67.8	25.4	6.8	2.1	34.3
SGMM + DNN Hybrid (Dan's) (it. 2)	combine_2 (2)	68.3	25.4	6.3	2.6	34.3
SGMM + DNN Hybrid (Dan's) (it. 3)	combine_2 (3)	68.4	25.4	6.3	2.5	34.1
SGMM + DNN Hybrid (Dan's) (it. 4)	combine_2 (4)	68.1	25.3	6.6	2.3	34.2
DNN Hybrid (Karel's)	dnn4_pretrain-dbn_dnn	67.2	24.8	8.0	1.7	34.5
DNN Hybrid (Karel's), sMBR training (it. 1)	dnn4_pretrain-dbn_dnn_smbr (1)	67.1	24.8	8.1	1.8	34.7
DNN Hybrid (Karel's), sMBR training (it. 6)	dnn4_pretrain-dbn_dnn_smbr (6)	67.6	24.9	7.5	2.1	34.5

Table 2a: results for the EVALITA 2016 ArtiPhon speaker dependent task in the 60-phones case.

Training & Decoding	%PCR	%SUB	%DEL	%INS	%PER
MonoPhone	mono	80.1	11.0	8.9	2.6	22.4
Delta + Delta-Deltas	tri1	85.4	7.7	6.9	2.6	17.2
LDA + MLTT	tri2	85.8	7.3	6.9	2.4	16.6
LDA + MLTT + SAT (SI)	tri3.si	85.2	7.5	7.3	2.7	17.6
LDA + MLTT + SAT	tri3	86.7	6.5	6.8	2.1	15.3
sgmm2_4: SGMM2	sgmm2_4	87.2	6.5	6.3	2.3	15.1
MMI + SGMM2 (iteration n.1)	sgmm2_4_mmi_b0.1	87.2	6.5	6.3	2.3	15.1
MMI + SGMM2 (iteration n.2)	sgmm2_4_mmi_b0.2	86.8	6.3	6.9	1.9	15.0
MMI + SGMM2 (iteration n.3)	sgmm2_4_mmi_b0.3	87.4	6.3	6.3	2.3	14.9
MMI + SGMM2 (iteration n.4)	sgmm2_4_mmi_b0.4	87.4	6.3	6.3	2.3	14.9
DNN Hybrid (Dan's)	tri4-nnet	82.8	8.5	8.8	2.4	19.7
SGMM + DNN Hybrid (Dan's) (it. 1)	combine_2 (1)	87.4	6.1	6.5	2.5	15.1
SGMM + DNN Hybrid (Dan's) (it. 2)	combine_2 (2)	87.4	6.1	6.5	2.5	15.1
SGMM + DNN Hybrid (Dan's) (it. 3)	combine_2 (3)	87.3	6.1	6.6	2.5	15.2
SGMM + DNN Hybrid (Dan's) (it. 4)	combine_2 (4)	87.3	6.1	6.6	2.5	15.2
DNN Hybrid (Karel's)	dnn4_pretrain-dbn_dnn	86.1	6.8	7.1	2.3	16.2
DNN Hybrid (Karel's), sMBR training (it. 1)	dnn4_pretrain-dbn_dnn_smbr (1)	86.1	6.8	7.2	2.3	16.2
DNN Hybrid (Karel's), sMBR training (it. 6)	dnn4_pretrain-dbn_dnn_smbr (6)	86.0	6.6	7.4	2.2	16.2

Table 2b: results for the EVALITA 2016 ArtiPhon speaker dependent task in the 29-phones case.

	Training & Decoding	%PCR	%SUB	%DEL	%INS	%PER
MonoPhone	mono	61,3	23.8	14.9	2.3	41.0
Delta + Delta-Deltas	tri1	66.8	21.3	11.9	3.7	36.9
LDA + MLTT	tri2	70.0	19.5	10.4	4.6	34.5
LDA + MLTT + SAT (SI)	tri3.si	70.2	18.3	11.5	2.2	32.0
LDA + MLTT + SAT	tri3	74.5	16.8	8.7	3.0	28.4
sgmm2_4: SGMM2	sgmm2_4	75.7	15.3	9.0	4.4	28.7
MMI + SGMM2 (iteration n.1)	sgmm2_4_mmi_b0.1	75.7	15.2	9.1	4.1	28.4
MMI + SGMM2 (iteration n.2)	sgmm2_4_mmi_b0.2	76.3	15.6	8.1	4.7	28.4
MMI + SGMM2 (iteration n.3)	sgmm2_4_mmi_b0.3	76.3	15.5	8.2	4.5	28.2
MMI + SGMM2 (iteration n.4)	sgmm2_4_mmi_b0.4	76.3	15.4	8.3	4.6	28.2
DNN Hybrid (Dan's)	tri4-nnet	70.7	17.3	12.0	3.7	31.8
SGMM + DNN Hybrid (Dan's) (it. 1)	combine_2 (1)	76.1	15.2	8.7	3.7	27.5
SGMM + DNN Hybrid (Dan's) (it. 2)	combine_2 (2)	76.2	15.0	8.8	3.6	27.4
SGMM + DNN Hybrid (Dan's) (it. 3)	combine_2 (3)	76.1	15.1	8.8	3.5	27.4
SGMM + DNN Hybrid (Dan's) (it. 4)	combine_2 (4)	76.2	15.2	8.6	3.4	27.1
DNN Hybrid (Karel's)	dnn4_pretrain-dbn_dnn	75.6	14.6	9.8	2.4	26.9
DNN Hybrid (Karel's), sMBR training (it. 1)	dnn4_pretrain-dbn_dnn_smbr (1)	75.3	14.6	10.2	2.3	27.1
DNN Hybrid (Karel's), sMBR training (it. 6)	dnn4_pretrain-dbn_dnn_smbr (6)	75.6	14.7	9.7	2.5	27.0

Table 3: results for the EVALITA 2016 ArtiPhone speaker independent task in the 29-phones case.

The results of the EVALITA 2016 ArtiPhon speaker independent experiment using the acoustic models trained on APASCI with the 29-phones are summarized in Table 3.

All the systems are built on top of MFCC, LDA, MLLT, fMLLR with CMN features[1] - see (Rath, et al., 2013) for all acronyms references - obtained from auxiliary GMM (Gaussian Mixture Model) models. At first, these 40-dimensional features are all stored to disk in order to simplify the training scripts.

Moreover MMI, BMMI, MPE and sMBR[2] training are all supported - see (Rath et al., 2013) for all acronyms references.

KALDI currently contains also two parallel implementations for DNN (Deep Neural Networks) training: "DNN Hybrid (Dan's)" (Kaldi, WEB-b), (Zhang et al., 2014), (Povey et al., 2015) and "DNN Hybrid (Karel's)" (Kaldi, WEB-a), (Vesely et al., 2013) in Table 3. Both of them are DNNs where the last (output) layer is a softmax layer whose output dimension equals the number of context-dependent states in the system (typically several thousand). The neural net is trained to predict the posterior probability of each context-dependent state. During decoding the output probabilities are divided by the prior probability of each state to form a "pseudo-likelihood" that is used in place of the state emission probabilities in the HMM(see Cosi et al. 2015, for a more detailed description).

The Phone Error Rate (PER) was considered for computing the score of the recognition process. The PER, which is defined as the sum of the deletion (DEL), substitution (SUB) and insertion (INS) percentage of phonemes in the ASR outcome text with respect to a reference transcription was computed by the use of the NIST software SCLITE (sctk-WEB).

The results shown in Table 3 refer to the various training and decoding experiments - see (Rath et al., 2013) for all acronyms references:

- MonoPhone (mono);
- Deltas + Delta-Deltas (tri1);
- LDA + MLLT (tri2);
- LDA + MLLT + SAT (tri3);
- SGMM2 (sgmm2_4);
- MMI + SGMM2 (sgmm2_4_mmi_b0.1-4);

[1] MFCC: Mel-Frequency Cepstral Coefficients; LDA: Linear Discriminant Analysis; MLTT: Maximum Likelihood Linear Transform; fMLLR: feature space Maximum Likelihood Linear Regression; CMN: Cepstral Mean Normalization.

[2] MMI: Maximum Mutual Information; BMMI: Boosted MMI; MPE: Minimum Phone Error; sMBR: State-level Minimum Bayes Risk

- Dan's Hybrid DNN (tri4-nnet),
- system combination, that is Dan's DNN + SGMM (combine_2_1-4);
- Karel's Hybrid DNN (dnn4_pretrain-dbn_dnn);
- system combination that is Karel's DNN + sMBR (dnn4_pretrain-dbn_dnn_1-6).

In the Table, SAT refers to the Speaker Adapted Training (SAT), i.e. train on fMLLR-adapted features. It can be done on top of either LDA+MLLT, or delta and delta-delta features.

If there are no transforms supplied in the alignment directory, it will estimate transforms itself before building the tree (and in any case, it estimates transforms a number of times during training). SGMM2 refers instead to Subspace Gaussian Mixture Models Training (Povey, 2009; Povey, et al. 2011). This training would normally be called on top of fMLLR features obtained from a conventional system, but it also works on top of any type of speaker-independent features (based on deltas+delta-deltas or LDA+MLLT).

5 Conclusions

As expected, due to the degraded clipped quality of the training and test audio signal, the 60-phones set is far from being realistic for obtaining optimum recognition performance even in the speaker dependent case (ArtiPhon training and test material).

On the contrary, if the reduced 29-phones set is used, the phone recognition performance is quite good and more than sufficient to build an effective ASR system if a language model could be incorporated.

Moreover, also in the speaker independent case (APASCI training material and ArtiPhon test material) the performance are not too bad even in these speech type and speaker mismatch conditions, thus confirming the effectiveness and the good quality of the system trained on APASCI material.

In these experiments, the DNNs results do not overcome those of the classic systems and we can hypothesize that this is due partially to the low quality of the signal, and also to the size of the corpus which is probably not sufficient to make the system learn all the variables characterizing the network. Moreover, the DNN architecture was not specifically tuned to the ArtiPhon data but instead the default KALDI architecture used in previous more complex speaker independent adult and children speech ASR experiments was simply chosen.

References

Angelini, B., Brugnara, F., Falavigna, D., Giuliani, D., Gretter, R., & Omologo, M., 1994. Speaker Independent Continuous Speech Recognition Using an Acoustic-Phonetic Italian Corpus. In Proc. of ICSLP, Yokohama, Japan, Sept. 1994, 1391-1394.

Bengio, Y., 2009. Learning Deep Architectures for AI, in Foundations and Trends in Machine Learning, Vol. 2, No. 1 (2009) 1-127.

Bourlard H:A. & Morgan N., 1994. Connectionist Speech Recognition: a Hybrid Approach, volume 247. Springer.

Canevari, C., Badino, L., Fadiga, L., 2015. A new Italian dataset of parallel acoustic and articulatory data, Proceedings of INTERSPEECH, Dresden, Germany, 2015, 2152-2156.

Cosi, P., & Hosom, J. P., 2000, High Performance General Purpose Phonetic Recognition for Italian, in *Proceedings of ICSLP 2000*, Beijing, 527-530, 2000.

Cosi, P. & Pellom, B., 2005. Italian Children's Speech Recognition For Advanced Interactive Literacy Tutors, in *Proceedings of INTERSPEECH 2005*, Lisbon, Portugal, 2201-2204, 2005.

Cosi, P., 2008. Recent Advances in Sonic Italian Children's Speech Recognition for Interactive Literacy Tutors, in *Proceedings of 1st Workshop On Child, Computer and Interaction (WOCCI-2008)*, Chania, Greece, 2008.

Cosi, P., 2009. On the Development of Matched and Mismatched Italian Children's Speech Recognition Systems, in *Proceedings of INTERSPEECH 2009*, Brighton, UK, 540-543, 2009.

Cosi, P., Nicolao, M., Paci, G., Sommavilla, G., & Tesser, F., 2014. Comparing Open Source ASR Toolkits on Italian Children Speech, in *Proceedings of Workshop On Child, Computer and Interaction (WOCCI-2014)*, Satellite Event of INTERSPEECH 2014, Singapore, September 19, 2014.

Cosi, P., Paci G., Sommavilla G., & Tesser F., 2015. KALDI: Yet another ASR Toolkit? Experiments on Italian Children Speech. In "Il farsi e il disfarsi del linguaggio. L'emergere, il mutamento e la patologia della struttura sonora del linguaggio", 2015

Dahl, G.E., Yu, D., Deng,L. & Acero, A., 2012. Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. IEEE Transactions on Audio, Speech, and Language Processing, Jan. 2012, 20(1):30–42.

Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N. & Kingsbury, B., 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition. IEEE Signal Processing Magazine, Nov. 2012, 29(6):82-97.

Kaldi-WEBa - Karel's DNN implementation: *http://KALDI.sourceforge.net/dnn1.html*

Kaldi-WEBb - Dan's DNN implementation: http://KALDI.sourceforge.net/dnn2.html.

Mohamed, A., Dahl, G.E. & Hinton, G., 2012. Acoustic Modeling Using Deep Belief Networks. IEEE Transactions on Audio, Speech, and Language Processing, Jan. 2012, 20(1):14-22.

Povey D., (2009). Subspace Gaussian Mixture Models for Speech Recognition, Tech. Rep. MSR-TR-2009-64, Microsoft Research, 2009.

Povey, D., Burget, L., Agarwal, M., Akyazi, P., Feng, K., Ghoshal, A:, Glembek, O., Goel, N.K., Karafiát, M., Rastrow, A., Rose, R.C., Schwarz, P., Thomas, S., (2011). The Subspace Gaussian Mixture Mode - A Structured Model for Speech Recognition, Computer Speech & Language, vol. 25, no. 2, pp. 404–439, April 2011.

Povey, D., Ghoshal, A. et al., 2001. The KALDI Speech Recognition Toolkit, in Proceedings of ASRU, 2011 (IEEE Catalog No.: CFP11SRW-USB).

Povey, D., Zhang, X., & Khudanpur, S., 2014. Parallel Training of DNNs with Natural Gradient and Parameter Averaging, in Proceedings of ICLR 2015, International Conference on Learning Representations (arXiv:1410.7455).

Rath, S. P., Povey, D., Vesely, K., & Cernocky, J., 2013. Improved feature processing for Deep Neural Networks, in Proceedings of INTERSPEECH 2013, 109-113.

sctk-WEB - Speech Recognition Scoring Toolkit https://www.nist.gov/itl/iad/mig/tools.

Serizel R., Giuliani D. (2014). Deep neural network adaptation for children's and adults' speech recognition. In Proceedings of ClicIt 2014, 1st Italian Conference on Computational Linguistics, Pisa, Italy, 2014.

Serizel R., Giuliani D. (2016). Deep-neural network approaches to speech recognition in heterogeneous groups of speakers including children, in Natural Language Engineering, April 2016.

Vesely, K., Ghoshal, A., Burget, L., & Povey, D., 2013. Sequence-discriminative training of deep neural networks, in Proceedings of INTERSPEECH 2013, 2345-2349.

Zhang, X, Trmal, J., Povey, D., & Khudanpur, S., 2014. Improving Deep Neural Network Acoustic Models Using Generalized Maxout Networks, in Proceedings of. ICASSP 2014, 215-219.

The EVALITA 2016 Event Factuality Annotation Task (FactA)

Anne-Lyse Minard[1,3], Manuela Speranza[1], Tommaso Caselli[2]
[1] Fondazione Bruno Kessler, Trento, Italy
[2] VU Amsterdam, the Netherlands
[3] Dept. of Information Engineering, University of Brescia, Italy
{speranza,minard}@fbk.eu
t.caselli@vu.nl

Abstract

English. This report describes the FactA (Event Factuality Annotation) Task presented at the EVALITA 2016 evaluation campaign. The task aimed at evaluating systems for the identification of the factuality profiling of events. Motivations, datasets, evaluation metrics, and post-evaluation results are presented and discussed.

Italiano. *Questo report descrive il task di valutazione FactA (Event Factaulity Annotation) presentato nell'ambito della campagna di valutazione EVALITA 2016. Il task si prefigge lo scopo di valutare sistemi automatici per il riconoscimento della fattualitá associata agli eventi in un testo. Le motivazioni, i dati usati, le metriche di valutazione, e risultati post-valutazione sono presentati e discussi.*

1 Introduction and Motivation

Reasoning about events plays a fundamental role in text understanding. It involves many aspects such as the identification and classification of events, the identification of event participants, the anchoring and ordering of events in time, and their factuality profiling.

In the context of the 2016 EVALITA evaluation campaign, we organized FactA (*Event Factuality Annotation*), the first evaluation exercise for factuality profiling of events in Italian. The task is a follow-up of Minard et al. (2015) presented in the track "Towards EVALITA 2016" at CLiC-it 2015. Factuality profiling is an important component for the interpretation of the events in discourse. Different inferences can be made from events which have not happened (or whose happening is probable) than from those which are described as factual. Many NLP applications such as Question Answering, Summarization, and Textual Entailment, among others, can benefit from the availability of this type of information.

Factuality emerges through the interaction of linguistic markers and constructions and its annotation represents a challenging task. The notion of factuality is strictly related to other research areas throughly explored in NLP, such as subjectivity, belief, hedging and modality (Wiebe et al., 2004; Prabhakaran et al., 2010; Medlock and Briscoe, 2007; Saurı et al., 2006). In this work, we adopted a notion of factuality which corresponds to the committed belief expressed by relevant sources towards the status of an event (Saurí and Pustejovsky, 2012). In particular, the factuality profile of events is expressed by the intersections of two axes: i.) certainty, which expresses a continuum which ranges from absolutely certain to uncertain; and ii.) polarity, which defines a binary distinction: affirmed (or positive) *vs.* negated (or negative).

In recent years, factuality profiling has been the focus of several evaluation exercises and shared tasks, especially for English, both in the newswire domain and in the biomedical domain. To mention the most relevant:

- the BioNLP 2009 Task 3 [1] and BioNLP 2011 Shared [2] Task aimed at recognizing if biomolecular events were affected by speculation or negation;
- the CoNLL 2010 Share Task [3] focused on hedge detection, i.e. identify speculated events, in biomedical texts;
- the ACE Event Detection and Recognition

[1] http://www.nactem.ac.uk/tsujii/GENIA/SharedTask/
[2] http://2011.bionlp-st.org
[3] http://rgai.inf.u-szeged.hu/index.php?lang=en&page=conll2010st

tasks [4] required systems to distinguish between asserted and non-asserted (e.g. hypothetical, desired, and promised) extracted events in news articles;

- the 2012 *SEM Shared Task on Resolving The Scope of Negation [5] focused one of its substasks on the identification of negated, i.e. counterfactual, events;
- the Event Nugget Detection task at TAC KBP 2015 Event Track [6] aimed at assessing the performance of systems in identifying events and their factual, or *realis*, value in news (Mitamura et al., 2015);
- the 2015 [7] and 2016 [8] SemEval Clinical TempEval tasks required systems to assign the factuality value (i.e. attributed modality and polarity) to the extracted events in clinical notes.

Finally recent work, such as the Richer Event Description annotation initiative,[9] has extended the annotation of factuality on temporal relations between pairs of events or pairs of events and temporal expressions as a specific task, independent from the factuality of the events involved, to represent claims about the certainty of the temporal relations themselves.

FactA provides the research community with new benchmark datasets and an evaluation environment to assess system performance concerning the assignment of factuality values to events. The evaluation is structured in two tasks: a Main Task, which focuses on the factuality profile of events in the newswire domain, and a Pilot Task, which addresses the factuality profiling of events expressed in tweets. To better evaluate system performance on factuality profiling and avoid the impact of errors from related subtasks, such as event identification, we restricted the task to the assignment of factuality values. Although as many as 13 teams registered for the task, none of those teams actually submitted any output. Nevertheless, we were able to run an evaluation following the evaluation campaign conditions for one system which was developed by one of the organizers, FactPro.

The remainder of the paper is organized as follows: the evaluation exercise is described in detail in Section 2, while the datasets are presented in Section 3. In Section 4 we describe the evaluation methodology and in Section 5 the results obtained with the FactPro system are illustrated. We conclude the paper in Section 6 with a discussion about the task and future work.

2 Task Description

Following Tonelli et al. (2014) and Minard et al. (2014), in FactA we represent factuality by means of three attributes associated with events,[10] namely *certainty*, *time*, and *polarity*. The FactA task consisted of taking as input a text in which the textual extent of events is given (i.e. gold standard data) and assign to the events the correct values for the three factuality attributes [11] according to the relevant source. In FactA, the relevant source is either the utterer (in direct speech, indirect speech or reported speech) or the author of the news (in all other cases). Systems do not have to provide the overall factuality value (FV): this is computed automatically on the basis of the *certainty*, *time*, and *polarity* attributes (see Section 2.2 for details).

2.1 Factuality Attributes

Certainty. *Certainty* relates to how sure the relevant source is about the mentioned event and admits the following three values: `certain` (e.g. 'rassegnato' in [1]), `non_certain` (e.g. 'usciti' in [2]), and `underspecified` (e.g. 'spiegazioni' in [3]).

1. *Smith ha rassegnato ieri le dimissioni; nomineranno il suo successore entro un mese. ("Smith resigned yesterday; they will appoint his replacement within a month.")*

[4] http://itl.nist.gov/iad/mig/tests/ace/

[5] http://ixa2.si.ehu.es/starsem/index.php\%3Foption=com_content&view=article&id=52&Itemid=60.html

[6] http://www.nist.gov/tac/2015/KBP/Event/index.html

[7] http://alt.qcri.org/semeval2015/task6/

[8] http://alt.qcri.org/semeval2016/task12/

[9] https://github.com/timjogorman/RicherEventDescription/blob/master/guidelines.md

[10] Based on the TimeML specifications (Pustejovsky et al., 2003), the term *event* is used as a cover term for situations that happen or occur, including predicates describing states or circumstances in which something obtains or holds true.

[11] Detailed instruction are reported in the FactA Annotation Guidelines available at http://facta-evalita2016.fbk.eu/documentation

2. *Probabilmente i ragazzi sono usciti di casa tra le 20 e le 21. ("The guys went probably out between 8 and 9 p.m.")*

3. *L'Unione Europea ha chiesto "spiegazioni" sulla strage di Beslan. ("The European Union has asked for an explanation about the massacre of Beslan.")*

Time. *Time* specifies the time when an event is reported to have taken place or is going to take place. Its values are `past/present` (for non-future events, e.g. 'capito' in [4]), `future` (for events that will take place, e.g. 'lottare' in [4]) or 'nomineranno' in [1]), and `underspecified` (e.g. 'verifica' in [5]).

4. *I russi hanno capito che devono lottare insieme. ("Russians have understood that they must fight together.")*

5. *Su 542 aziende si hanno i dati definitivi mentre per le altre 38 si è tuttora in fase di verifica. ("They have the final data for 542 companies while for the other 38 it is under validation.")*

Polarity. *Polarity* captures whether an event is affirmed or negated and, consequently, it can be either `positive` (e.g. 'rassegnato' in [1]) or `negative` (e.g. 'nominato' in [6]); when there is not enough information available to detect the *polarity* of an event mention, its value is `underspecified` (e.g. 'scompone' in [7]).

6. *Non ha nominato un amministratore delegato. ("He did not appoint a CEO.")*

7. *Se si scompone il dato sul nero, si vede che il 23% è dovuto a lavoratori residenti in provincia. ("If we analyze the data about the black market labor, we can see that 23% is due to workers resident in the province.")*

Event mentions in texts can be used to refer to events that do not correlate with a real situation in the world (e.g. 'parlare' in [8]). For these event mentions, participant systems are required to leave the value of all three attributes empty.

8. *Guardate, penso che sia prematuro parlare del nuovo preside. ("Well, I think it is too early to talk about the new dean.")*

2.2 Factuality Value

The combination of the *certainty*, *time*, and *polarity* attributes described above determines the factuality value (FV) of an event with respect to the relevant source.

As shown in Table 1, the FV can assume five values: i.) `factual`; ii.) `counterfactual`; iii.) `non-factual`; iv.) `underspecified`; and v.) `no factuality` (no fact). We illustrate in Table 1 the full set of valid combinations of the attribute values and the corresponding FV.

A `factual` value is assigned if an event has the following configuration of attributes:

- *certainty*: `certain`
- *time*: `past/present`
- *polarity*: `positive`

For instance, the event 'rassegnato' [*resigned*] in [1] will qualify as a `factual` event. On the other hand, a change in the *polarity* attribute, i.e. `negative`, will give rise to a `counterfactual` FV, like for instance the event 'nominato' [*appointed*] in [6].

`Non-factual` events depend on the values of the *certainty* and *time* attributes. In particular, a `non-factual` value is assigned if either of the two cases below occur, namely:

- *certainty*: `non_certain`; or
- *time*: `future`

This is the case for the event 'lottare' [*fight*] in [4], where *time* is `future`, or the event 'usciti' [*went out*] in [2] where *certainty* is `non_certain`.

The event FV is `underspecified` if at least one between *certainty* and *time* is `underspecified`, independently of the *polarity* value, like for instance in the case of 'verifica' [*validation*] in [5].

Finally, if the three attributes have no value, FV is `no factuality` (e.g. 'parlare' [*discuss*] in [8]).

3 Dataset Description

We made available an updated version of Fact-Ita Bank (Minard et al., 2014) as training data to participants. This consists of 169 documents selected from the Ita-TimeBank (Caselli et al., 2011) and

[12]The number of tokens for the pilot test is computed after the tokenizaion, i.e. the hashtags and aliases can be split in more than one token and the emoji are composed by several tokens.

Certainty	Time	Polarity	FV
certain	past/pres.	positive	factual
certain	past/pres.	negative	counterfact.
non_cert.	*any value*	*any value*	non-fact.
any value	future	*any value*	non-fact.
certain	undersp.	*any value*	underspec.
undersp.	past/pres.	*any value*	underspec.
undersp.	undersp.	*any value*	underspec.
-	-	-	no fact.

Table 1: Possible combinations of factuality attributes.

first released for the EVENTI task at EVALITA 2014.[13] Fact-Ita Bank contains annotations for 6,958 events (see Table 2 for more details) and is distributed with a CC-BY-NC license.[14]

As test data for the Main Task we selected the Italian section of the NewsReader MEANTIME corpus (Minard et al., 2016), a corpus of 120 Wikinews articles annotated at multiple levels. The Italian section is called WItaC, the NewsReader Wikinews Italian Corpus (Speranza and Minard, 2015) , and consists of 15,676 tokens (see Table 2).

As test data for the Pilot Task we annotated 301 tweets with event factuality, representing a subsection of the test set of the EVALITA 2016 SENTIPOLC task (Barbieri et al., 2016) (see Table 2).

Training and test data, both for the Main and the Pilot Tasks, are in the CAT (Content Annotation Tool) (Bartalesi Lenzi et al., 2012) labelled format. This is an XML-based stand-off format where different annotation layers are stored in separate document sections and are related to each other and to source data through pointers.

4 Evaluation

Participation in the task consisted of providing only the values for the three factuality attributes (*certainty*, *time*, *polarity*), while the FV score was to be computed through the FactA scorer on top of these values.

The evaluation is based on the micro-average F1 score of the FVs, which is equivalent to the accuracy in this task as all events should receive a FV (i.e. the total numbers of False Positives and False Negatives over the classes are equal). In addition to this, an evaluation of the performance of the systems on the single attributes (using micro-average F1 score, equivalent to the accuracy) will be provided as well. We consider this type of evaluation to be more informative than the one based on the single FV because it will provide evidence of systems' ability to identify the motivations for the assignment of a certain factuality value. To clarify this point, consider the case of an event with FV `non-factual` (certainty `non_certain`, time `past/present` and polarity `positive`). A system might correctly identify that the FV of the event is `non-factual` because certainty is `non_certain`, or erroneously identify that time is `future`.

5 System Results

Unfortunately no participants took part in the FactA task. However, we managed to run an evaluation test with a system for event factuality annotation in Italian, FactPro, developed by one of the organizers and respecting the evaluation campaign conditions. The system was evaluated against both gold standard, i.e. the Main and Pilot tasks. In this section we describe this system and the results obtained on the FactA task.

5.1 FactPro module

FactPro is a module of the TextPro NLP pipeline [15] (Pianta et al., 2008). It has been developed by Anne-Lyse Minard in collaboration with Federico Nanni as part of an internship.

Event Factuality annotation is performed in FactPro in three steps: (1) detection of the polarity of an event, (2) identification of the certainty of an event and (3) identification of the semantic time. These three steps are based on a machine learning approach, using Support Vector Machines algorithm, and are taken as text chunking tasks in which events have to be classified in different classes. For each step a multi-class classification model is built using the text chunker Yamcha (Kudo and Matsumoto, 2003).

FactPro requires the following pre-processes: sentence splitting, tokenization, morphological analysis, lemmatization, PoS tagging, chunking, and event detection and classification. As the data provided for FactA consist of texts already split into sentences, tokenized and annotated with events, the steps of sentence splitting, tokenization and event

[13] `https://sites.google.com/site/eventievalita2014/home`

[14] `http://hlt-nlp.fbk.eu/technologies/fact-ita-bank`

[15] `http://textpro.fbk.eu`

	training set (main) Fact-Ita Bank	**test set (main)** MEANTIME	**test set (pilot)** (tweets)
tokens[12]	65,053	15,676	4,920
sentences	2,723	597	301
events	6,958	1,450	475
certainty			
certain	5,887	1,246	326
non certain	813	133	53
underspecified	204	53	43
time			
past/present	5,289	1,026	263
future	1,560	318	113
underspecified	55	88	46
polarity			
positive	6,474	1,363	381
negative	378	45	27
underspecified	52	24	14
FV			
factual	4,831	978	225
counterfactual	262	32	15
non-factual	1,700	327	126
underspecified	111	95	56
no_factuality	54	18	53

Table 2: Corpora statistics

detection and classification are not performed for these experiments.

Each classifier makes use of different features: lexical, syntactic and semantic. They are described in the remainder of the section. For the detection of polarity and certainty, FactPro makes use of trigger lists which have been built manually using the training corpus.

- Polarity features:
  - For all tokens: token's lemma, PoS tags, whether it is a polarity trigger (list manually built);
  - If the token is part of an event: presence of polarity triggers before it, their number, the distance to the closest trigger, and whether the event is part of a conditional construction;
  - The polarity value tagged by the classifier for the two preceding tokens.
- Certainty features:
  - For all tokens: token's lemma, flat constituent (noun phrase or verbal phrase), whether it is a modal verb, whether it is a certainty trigger (list manually built);
  - If the token is part of an event: the event class (It-TimeML classes), presence of a modal before and its value, and whether the event is part of a conditional construction;
  - The certainty value tagged by the classifier for the two preceding tokens.
- Time features:
  - For all tokens: token's lemma and whether it is a preposition;
  - If the token is part of an event: tense and mood of the verb before, presence of a preposition before, event's polarity and certainty;
  - If the token is a verb: its tense and mood;
  - The time value tagged by the classifier for the three preceding tokens.

Each token is represented using these features as well as some of the features of the previous tokens and of the following ones. We have defined the set of features used by each classifier performing several evaluations on a subsection of the Fact-Ita Bank corpus.

task	system	polarity	certainty	time	3 attributes	Factuality Value
main	baseline	0.94	0.86	0.71	0.67	0.67
main	FactPro	0.92	0.83	0.74	0.69	**0.72**
pilot	baseline	0.80	0.69	0.55	0.47	0.47
pilot	FactPro	0.79	0.66	0.60	0.51	**0.56**

Table 3: Evaluation of FactPro against the baseline (accuracy)

5.2 Results

Table 3 shows the results of FactPro for the two tasks of FactA against a baseline. The baseline system annotates all events as `factual` (the predominant class), i.e. being `certain`, `positive` and `past/present`. The performance of FactPro on the Main Task is 0.72 when evaluating the Factuality Value assignment and 0.69 on the combination of the three attributes, and on the Pilot Task 0.56 and 0.51 respectively. On these two tasks FactPro performs better than the baseline. It has to be noted that we ran FactPro on the pilot test set without any adaptation of the different processes.

In Table 4 we present the F1-score obtained for each value of the three attributes as well for each Factuality Value. We can observe that FactPro does not perform well on the identification of the `underspecified` values and on the detection of events that do not have a factuality value (`no fact`).

5.3 Error Analysis of FactPro

We can observe from Table 3 that FactPro performs better for the detection of polarity and certainty than for the identification of time. One reason is the predominance of one value for the polarity and certainty attributes, and of two values for time. For example, in the training corpus, 94% of the events have a polarity `positive` and 86% are `certain`, whereas 71% of the events are `past/present` and 22% are `future`.

An extensive error analysis on the output of the systems for the three attributes was conducted. As for the polarity attribute, the error analysis showed that the system's failure to detect negated events is not mainly due to a sparseness of negated events in the training data, but it mainly concerns the negation scope, whereas when the system missed a negative event it was mainly due to the incompleteness of the trigger lists (e.g. *mancata* in *dopo la mancata approvazione* is a good trigger for polarity `negative` but it is absent from the trigger list).

The detection of `non_certain` events works well when the event is preceded by a verb at the conditional and when it is part of an infinitive clause introduced by *per*. However when the uncertainty of an event is expressed by the semantics of previous words (e.g. *salvataggio* in *il piano di salvataggio*) the system makes errors.

With respect to the annotation of `future` events, the observations are similar to those for `non_certain` events. Indeed, future events are well recognized by the system when they are part of an infinitive clause introduced by the preposition *per* as well as when their tense is future.

Finally, we observed that FactPro makes a lot of errors when the annotation of the factuality of nominal events is concerned. In the Main Task it correctly identified the FV of 81% of the verbal events and only 61% of the nominal events.

6 Conclusion and Future Work

The lack of participants in the task limits the discussion of the results to the in-house developed system. The main reason for the lack of participation to FactA, according to the outcome of a questionnaire organized by the 2016 EVALITA chairs, was that the participants gave priority to other EVALITA tasks. However, FactA achieves two main results: i.) setting state-of-the-art results for the factuality profiling of events in two text types in Italian, namely news articles and tweets; and ii.) making available to the community a new benchmark corpus and standardized evaluation environment for comparing systems' performance and facilitating replicability of results.

The test data used for the Main Task consists of the Italian section of the MEANTIME corpus (Minard et al., 2016). MEANTIME contains the same documents aligned in English, Italian, Spanish and Dutch, thus making available a multilingual environment for cross-language evaluation of the factuality profiling of events. Furthermore, within the NewsReader project, a module for event factuality annotation has been implemented and evaluated against the English section of the MEANTIME

	polarity			certainty			time		
task	pos.	neg.	undersp.	cert.	non_cert.	undersp.	past/pres.	future	undersp.
main	0.96	0.68	0.00	0.91	0.42	0.10	0.84	0.54	0.00
pilot	0.88	0.69	0.00	0.80	0.35	0.18	0.73	0.50	0.00

	FV				
task	factual	counterfact.	non-fact.	undersp.	no fact.
main	0.83	0.62	0.55	0.02	0.00
pilot	0.72	0.39	0.50	0.03	0.29

Table 4: FactPro results on the single attribute and on the different factuality value (F1 score)

corpus (Agerri et al., 2015). The evaluation was performed in a different way than in FactA, in particular no gold events were provided as input to the system, so the evaluation of factuality was done only for the events correctly identified by the event detection module. The system obtained an accuracy of 0.88, 0.86 and 0.59 for polarity, certainty, and time, respectively.

The Pilot task was aimed at evaluating how well systems built for standard language perform on social media texts, and at making available a set of tweets annotated with event mentions (following TimeML definition of events) and their factuality value. The pilot data are shared between three other tasks of EVALITA 2016 (PoSTWITA, NEEL-IT and SENTIPOLC), which contributed to the creation of a richly annotated corpus of tweets to be used for future cross-fertilization tasks. Finally, the annotation of tweets raised new issues for factuality annotation because tweets contain a lot of imperatives and interrogatives that are generally absent from news and for which the factuality status is not obvious (e.g. *Ordini solo quello che ti serve*).

The results obtained by FactPro, as reported in Table 3 and Table 4, show that i.) the system is able to predict with pretty high accuracy the FV on events in the news domain and with a lower but good score the factuality of events in tweets; ii.) the difference in performance between the news and tweet text types suggest that specific training set data may be required to address the peculiarities of tweets' language; iii.) the F1 scores for the certainty, polarity and time attributes clearly indicate areas of improvements and also contribute to a better understanding of the system's results; iv.) the F1 scores on the FV suggest that extending the training data with tweets could also benefit the identification of values which are not frequent in the news domain, such as `no_fact`.

Future work will aim at re-running the task from raw text and developing specific modules for the factuality of events according to the text types where they occur. Finally, we will plan to run a cross-fertilization task concerning temporal ordering and anchoring of events and factauality profiling.

Acknowledgments

This work has been partially supported by the EU NewsReader Project (FP7-ICT-2011-8 grant 316404) and the NWO Spinoza Prize project Understanding Language by Machines (sub-track 3).

References

Rodrigo Agerri, Itziar Aldabe, Zuhaitz Beloki, Egoitz Laparra, German Rigau, Aitor Soroa, Marieke van Erp, Antske Fokkens, Filip Ilievski, Ruben Izquierdo, Roser Morante, Chantal van Son, Piek Vossen, and Anne-Lyse Minard. 2015. Event Detection, version 3. Technical Report D4-2-3, VU Amsterdam. `http://kyoto.let.vu.nl/newsreader_deliverables/NWR-D4-2-3.pdf`.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Valentina Bartalesi Lenzi, Giovanni Moretti, and Rachele Sprugnoli. 2012. CAT: the CELCT Annotation Tool. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12)*, pages 333–338, Istanbul, Turkey, May. European Language Resources Association (ELRA).

Tommaso Caselli, Valentina Bartalesi Lenzi, Rachele Sprugnoli, Emanuele Pianta, and Irina Prodanof. 2011. Annotating Events, Temporal Expressions

and Relations in Italian: the It-TimeML Experience for the Ita-TimeBank. In *Linguistic Annotation Workshop*, pages 143–151.

Taku Kudo and Yuji Matsumoto. 2003. Fast Methods for Kernel-based Text Analysis. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 24–31, Stroudsburg, PA, USA.

Ben Medlock and Ted Briscoe. 2007. Weakly supervised learning for hedge classification in scientific literature. In *ACL*, volume 2007, pages 992–999. Citeseer.

Anne-Lyse Minard, Alessandro Marchetti, and Manuela Speranza. 2014. Event Factuality in Italian: Annotation of News Stories from the Ita-TimeBank. In *Proceedings of CLiC-it 2014, First Italian Conference on Computational Linguistic*.

Anne-Lyse Minard, Manuela Speranza, Rachele Sprugnoli, and Tommaso Caselli. 2015. FacTA: Evaluation of Event Factuality and Temporal Anchoring. In *Proceedings of the Second Italian Conference on Computational Linguistics CLiC-it 2015*.

Anne-Lyse Minard, Manuela Speranza, Ruben Urizar, Begoa Altuna, Marieke van Erp, Anneleen Schoen, and Chantal van Son. 2016. Meantime, the newsreader multilingual event and time corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may.

Teruko Mitamura, Yukari Yamakawa, Susan Holm, Zhiyi Song, Ann Bies, Seth Kulick, and Stephanie Strassel. 2015. Event Nugget Annotation: Processes and Issues. In *Proceedings of the 3rd Workshop on EVENTS at the NAACL-HLT*, pages 66–76.

Emanuele Pianta, Christian Girardi, and Roberto Zanoli. 2008. The TextPro Tool Suite. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco.

Vinodkumar Prabhakaran, Owen Rambow, and Mona Diab. 2010. Automatic committed belief tagging. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1014–1022. Association for Computational Linguistics.

James Pustejovsky, José M. Castaño, Robert Ingria, Roser Saurí, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. 2003. TimeML: Robust Specification of Event and Temporal Expressions in Text. In *New Directions in Question Answering*, pages 28–34.

Roser Saurí and James Pustejovsky. 2012. Are you sure that this happened? Assessing the factuality degree of events in text. *Computational Linguistics*, 38(2):261–299.

Roser Saurı, Marc Verhagen, and James Pustejovsky. 2006. Annotating and recognizing event modality in text. In *Proceedings of 19th International FLAIRS Conference*.

Manuela Speranza and Anne-Lyse Minard. 2015. Cross-language projection of multilayer semantic annotation in the NewsReader Wikinews Italian Corpus (WItaC). In *Proceedings of CLiC-it 2015, Second Italian Conference on Computational Linguistic*.

Sara Tonelli, Rachele Sprugnoli, Manuela Speranza, and Anne-Lyse Minard. 2014. NewsReader Guidelines for Annotation at Document Level. Technical Report NWR2014-2-2, Fondazione Bruno Kessler. `http://www.newsreader-project.eu/files/2014/12/NWR-2014-2-2.pdf`.

Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. 2004. Learning subjective language. *Computational linguistics*, 30(3):277–308.

Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) Task

Pierpaolo Basile[1] and **Annalina Caputo**[2] and **Anna Lisa Gentile**[3] and **Giuseppe Rizzo**[4]
[1] Department of Computer Science, University of Bari Aldo Moro, Bari (Italy)
[2] ADAPT Centre, Trinity Collge Dublin, Dublin (Ireland)
[3] University of Mannheim, Mannheim (Germany)
[4] Istituto Superiore Mario Boella, Turin (Italy)
[1]pierpaolo.basile@uniba.it
[2]annalina.caputo@adaptcentre.ie
[3]annalisa@informatik.uni-mannheim.de
[4]giuseppe.rizzo@ismb.it

Abstract

English. This report describes the main outcomes of the 2016 Named Entity rEcognition and Linking in Italian Tweet (NEEL-IT) Challenge. The goal of the challenge is to provide a benchmark corpus for the evaluation of entity recognition and linking algorithms specifically designed for noisy and short texts, like tweets, written in Italian. The task requires the correct identification of entity mentions in a text and their linking to the proper named entities in a knowledge base. To this aim, we choose to use the canonicalized dataset of DBpedia 2015-10. The task has attracted five participants, for a total of 15 runs submitted.

Italiano. *In questo report descriviamo i principali risultati conseguiti nel primo task per la lingua Italiana di Named Entity rEcognition e Linking in Tweet (NEEL-IT). Il task si prefigge l'obiettivo di offrire un framework di valutazione per gli algoritmi di riconoscimento e linking di entità a nome proprio specificamente disegnati per la lingua italiana per testi corti e rumorosi, quali i tweet. Il task si compone di una fase di riconoscimento delle menzioni di entità con nome proprio nel testo e del loro successivo collegamento alle opportune entità in una base di conoscenza. In questo task abbiamo scelto come base di conoscenza la versione canonica di DBpedia 2015. Il task ha attirato cinque partecipanti per un totale di 15 diversi run.*

1 Introduction

Tweets represent a great wealth of information useful to understand recent trends and user behaviours in real-time. Usually, natural language processing techniques would be applied to such pieces of information in order to make them machine-understandable. Named Entity rEcongition and Linking (NEEL) is a particularly useful technique aiming aiming to automatically annotate tweets with named entities. However, due to the noisy nature and shortness of tweets, this technique is more challenging in this context than elsewhere. International initiatives provide evaluation frameworks for this task, e.g. the Making Sense of Microposts workshop (Dadzie et al., 2016) hosted the 2016 NEEL Challenge (Rizzo et al., 2016), or the W-NUT workshop at ACL 2015 (Baldwin et al., 2015), but the focus is always and strictly on the English language. We see an opportunity to (i) encourage the development of language independent tools for for Named Entity Recognition (NER) and Linking (NEL) systems and (ii) establish an evaluation framework for the Italian community. NEEL-IT at EVALITA has the vision to establish itself as a reference evaluation framework in the context of Italian tweets.

2 Task Description

NEEL-IT followed a setting similar to NEEL challenge for English Micropost on Twitter (Rizzo et al., 2016). The task consists of annotating each named entity mention (like *people*, *locations*, *organizations*, and *products*) in a text by linking it to a knowledge base (DBpedia 2015-10).

Specifically, each task participant is required to:

1. Recognize and typing each entity mention that appears in the text of a tweet;

Table 1: Example of annotations.

id	begin	end	link	type
288...	0	18	NIL	Product
288...	73	86	http://dbpedia.org/resource/Samsung_Galaxy_Note_II	Product
288...	89	96	http://dbpedia.org/resource/Nexus_4	Product
290...	1	15	http://dbpedia.org/resource/Carlotta_Ferlito	Person

2. Disambiguate and link each mention to the canonicalized DBpedia 2015-10, which is used as referent Knowledge Base. This means that if an entity is present in the Italian DBpedia but not in the canonicalized version, this mention should be tagged as NIL. For example, the mention *Agorà* can only be referenced to the Italian DBpedia entry Agorà <*programma televisivo*>[1], but this entry has no correspondence into the canonicalized version of DBpedia. Then, it has been tagged as a NIL instance.

3. Cluster together the non linkable entities, which are tagged as NIL, in order to provide a unique identifier for all the mentions that refer to the same named entity.

In the annotation process, a named entity is a string in the tweet representing a proper noun that: 1) belongs to one of the categories specified in a taxonomy and/or 2) can be linked to a DBpedia concept. This means that some concepts have a NIL DBpedia reference[2].

The taxonomy is defined by the following categories:

Thing languages, ethnic groups, nationalities, religions, diseases, sports, astronomical objects;

Event holidays, sport events, political events, social events;

Character fictional character, comics character, title character;

Location public places, regions, commercial places, buildings;

Organization companies, subdivisions of companies, brands, political parties, government bodies, press names, public organizations, collection of people;

Person people's names;

Product movies, tv series, music albums, press products, devices.

From the annotation are excluded the preceding article (like il, lo, la, etc.) and any other prefix (e.g. Dott., Prof.) or post-posed modifier. Each participant is asked to produce an annotation file with multiple lines, one for each annotation. A line is a tab separated sequence of tweet id, start offset, end offset, linked concept in DBpedia, and category. For example, given the tweet with id 288976367238934528:

Chameleon Launcher in arrivo anche per smartphone: video beta privata su Galaxy Note 2 e Nexus 4: Chameleon Laun...

the annotation process is expected to produce the output as reported in Table 1.

The annotation process is also expected to link Twitter mentions (@) and hashtags (#) that refer to a named entities, like in the tweet with id 290460612549545984:

@CarlottaFerlito io non ho la forza di alzarmi e prendere il libro! Help me

the correct annotation is also reported in Table 1.

Participants were allowed to submit up to three runs of their system as TSV files. We encourage participants to make available their system to the community to facilitate reuse.

3 Corpus Description and Annotation Process

The NEEL-IT corpus consists of both a development set (released to participants as training set) and a test set. Both sets are composed by two TSV files: (1) the tweet id file, this is a list of all tweet ids used for training; (2) the gold standard,

[1] `http://it.dbpedia.org/resource/AgorÃă\_(programma\_televisivo)`

[2] These concepts belong to one of the categories but they have no corresponding concept in DBpedia

containing the annotations for all the tweets in the development set following the format showed in Table 1.

The development set was built upon the dataset produced by Basile et al. (2015). This dataset is composed by a sample of 1,000 tweets randomly selected from the TWITA dataset (Basile and Nissim, 2013). We updated the gold standard links to the canonicalized DBpedia 2015-10. Furthermore, the dataset underwent another round of annotation performed by a second annotator in order to maximize the consistency of the links. Tweets that presented some conflicts were then resolved by a third annotator.

Data for the test set was generated by randomly selecting 1,500 tweets from the SENTIPOLC test data (Barbieri et al., 2016). From this pool, 301 tweets were randomly chosen for the annotation process and represents our Gold Standard (GS). This sub-sample was choose in coordination with the task organisers of SENTIPOLC (Barbieri et al., 2016), POSTWITA (Tamburini et al., 2016) and FacTA (Minard et al., 2016b) with the aim of providing a unified framework for multiple layers of annotations.

The tweets were split in two batches, each of them was manually annotated by two different annotators. Then, a third annotator intervened in order to resolve those debatable tweets with no exact match between annotations. The whole process has been carried out by exploiting BRAT[3] web-based tool (Stenetorp et al., 2012).

Table 2 reports some statistics on the two sets: in both the most represented categories are "Person", "Organization" and "Location". "Person" is also the most populated category among the NIL instances, along to "Organization" and "Product". In the development set, the least represented category is "Character" among the NIL instances and both "Thing" and "Event" between the linked ones. A different behaviour can be found in the test set where the least represented category is "Thing" in both NIL and linked instances.

Table 2: Datasets Statistics.

Stat.	Dev. Set	Test Set
# tweets	1,000	301
# tokens	14,242	4,104
# hashtags	250	108
# mentions	624	181
Mean token per tweet	14.24	13.65
# NIL Thing	14	3
# NIL Event	9	7
# NIL Character	4	5
# NIL Location	6	9
# NIL Organization	49	19
# NIL Person	150	76
# NIL Product	43	12
# Thing	6	0
# Event	6	12
# Character	12	2
# Location	116	70
# Organization	148	56
# Person	173	61
# Product	65	25
# NIL instances	275	131
# Entities	526	357

4 Evaluation Metrics

Each participant was asked to submit up to three different run. The evaluation is based on the following three metrics:

STMM (*Strong_Typed_Mention_Match*). This metrics evaluates the micro average F-1 score for all annotations considering the mention boundaries and their types. This is a measure of the tagging capability of the system.

SLM (*Strong_Link_Match*). This metrics is the micro average F-1 score for annotations considering the correct link for each mention. This is a measure of the linking performance of the system.

MC (*Mention_Ceaf*). This metrics, also known as Constrained Entity-Alignment F-measure (Luo, 2005), is a clustering metric developed to evaluate clusters of annotations. It evaluates the F-1 score for both NIL and non-NIL annotations in a set of mentions.

The final score for each system is a combination of the aforementioned metrics and is computed as follows:

$$score = 0.4 \times MC + 0.3 \times STMM + 0.3 \times SLM. \quad (1)$$

All the metrics were computed by using the TAC KBP scorer[4].

[3] http://brat.nlplab.org/

[4] https://github.com/wikilinks/neleval/

5 Systems Description

The task was well received by the NLP community and was able to attract 17 participants who expressed their interest in the evaluation. Five groups participated actively to the challenge by submitting their system results, each group presented three different runs, for a total amount of 15 runs submitted. In this section we briefly describe the methodology followed by each group.

5.1 UniPI

The system proposed by the University of Pisa (Attardi et al., 2016) exploits word embeddings and a bidirectional LSTM for entity recognition and linking. The team produced also a training dataset of about 13,945 tweets for entity recognition by exploiting active learning, training data taken from the PoSTWITA task (Tamburini et al., 2016) and manual annotation. This resource, in addition to word embeddings built on a large corpus of Italian tweets, is used to train a bidirectional LSTM for the entity recognition step. In the linking step, for each Wikipedia page its abstract is extracted and the average of the word embeddings is computed. For each candidate entity in the tweet, the word embedding for a context of words of size c before and after the entity is created. The linking is performed by comparing the mention embedding with the DBpedia entity whose lc2 distance is the smallest among those entities whose abstract embeddings were computed at the previous step. The Twitter mentions were resolved by retrieving the real name with the Twitter API and looking up in a gazetteer in order to identify the Person-type entities.

5.2 MicroNeel

MicroNeel (Corcoglioniti et al., 2016) investigates the use on microposts of two standard NER and Entity Linking tools originally developed for more formal texts, namely Tint (Palmero Aprosio and Moretti, 2016) and The Wiki Machine (Palmero Aprosio and Giuliano, 2016). Comprehensive tweet preprocessing is performed to reduce noisiness and increase textual context. Existing alignments between Twitter user profiles and DBpedia entities from the Social Media Toolkit (Nechaev et al., 2016) resource are exploited to annotate user mentions in the tweets. Rule-based and supervised (SVM-based) techniques are investigated to merge annotations from different tools and solve possible conflicts. All the resources listed as follows were employed in the evaluation:

- The Wiki Machine (Palmero Aprosio and Giuliano, 2016): an open source entity linking for Wikipedia and multiple languages.
- Tint (Palmero Aprosio and Moretti, 2016): an open source suite of NLP modules for Italian, based on Stanford CoreNLP, which supports named entity recognition.
- Social Media Toolkit (SMT) (Nechaev et al., 2016): a resource and API supporting the alignment of Twitter user profiles to the corresponding DBpedia entities.
- Twitter ReST API[5]: a public API for retrieving Twitter user profiles and tweet metadata.
- Morph-It! (Zanchetta and Baroni, 2005): a free morphological resource for Italian used for preprocessing (true-casing) and as source of features for the supervised merging of annotations.
- tagdef[6]: a website collecting user-contributed descriptions of hashtags.
- list of slang terms from Wikipedia[7].

5.3 FBK-HLT-NLP

The system proposed by the FBK-HLT-NLP team (Minard et al., 2016a) follows 3 steps: entity recognition and classification, entity linking to DBpedia and clustering. Entity recognition and classification is performed by the EntityPro module (included in the TextPro pipeline), which is based on machine learning and uses the SVM algorithm. Entity linking is performed using the named entity disambiguation module developed within the NewsReader and based on DBpedia Spotlight. The FBK team exploited a specific resource to link the Twitter profiles to DBpedia: the Alignments dataset. The clustering step is string-based, i.e. two entities are part of the same cluster if they are equal.

wiki/Evaluation

[5] https://dev.twitter.com/rest/public

[6] https://www.tagdef.com/

[7] https://it.wikipedia.org/wiki/Gergo_di_Internet

Moreover, the FBK team exploits active learning for domain adaptation, in particular to adapt a general purpose Named Entity Recognition system to a specific domain (tweets) by creating new annotated data. In total they have annotated 2,654 tweets.

5.4 Sisinflab

The system proposed by Sisinflab (Cozza et al., 2016) faces the neel-it challenge through an ensamble approach that combines unsupervised and supervised methods. The system merges results achieved by three strategies:

1. DBpedia Spotlight for span and URI detection plus SPARQL queries to DBpedia for type detection;

2. Stanford CRF-NER trained with the challenge train corpus for span and type detection and DBpedia lookup for URI detection;

3. DeepNL-NER, a deep learning classifier trained with the challenge train corpus for span and type detection, it exploits ad-hoc gazetteers and word embedding vectors computed with word2vec trained over the Twita dataset[8] (a subset of 12,000,000 tweets). DBpedia is used for URI detection.

Finally, the system computes NIL clusters for those mentions that do not match with an entry in DBpedia, by grouping in the same cluster entities with the same text (no matter the case). The Sisinflab team submitted three runs combining the previous strategies, in particular: run1) combines (1), (2) and (3); run2 involves strategies (1) and (3); run3 exploits strategies (1) and (2).

5.5 UNIMIB

The system proposed by the UNIMIB team (Cecchini et al., 2016) is composed of three steps: 1) Named Entity Recognition using Conditional Random Fields (CRF); 2) Named Entity Linking by considering both Supervised and Neural-Network Language models and 3) NIL clustering by using a graph-based approach. In the first step two kinds of CRF are exploited: 1) a simple CRF on the training data and 2) CRF+Gazetteers, in this configuration the model has been induced by exploiting several gazetteers, i.e. products, organizations, persons, events and characters. Two strategies are adopted for the linking. A decision strategy is used to select the best link by exploiting a large set of supervised methods. Then, word embeddings built on Wikipedia are used to compute a similarity measure used to select the best link for a list of candidate entities. NIL clustering is performed by a graph-based approach; in particular, a weighted indirect co-occurrence graph where an edge represents the co-occurrence of two terms in a tweet is built. The ensuing word graph was then clustered using the MaxMax algorithm.

[8] http://www.let.rug.nl/basile/files/proc/

6 Results

The performance of the participant systems were assessed by exploiting the final score measure presented in Eq. 1. This measure combines the three different aspects evaluated during the task, i.e. the correct tagging of the mentions (**STMM**), the proper linking to the knowledge base (**SLM**), and the clustering of the NIL instances (**MC**). Results of the evaluation in terms of the final score are reported in Table 3.

The best result was reported by *Uni.PI.3*, this system obtained the best final score of 0.5034 with an improvement with respect to the *Uni.PI.1* (second classified) of $+1.27$. The difference between these two runs lays on the different vector dimension (200 in *Uni.PI.3* rather than 100 in *Uni.Pi.1*) combined with the use of Wikipedia embeddings and a specific training set for geographical entities (*Uni.PI.3*) rather than a mention frequency strategy for disambiguation (*Uni.PI.1*). *MicroNeel.base* and *FBK-HLT-NLP* obtain remarkable results very close to the best system. Indeed, *MicroNeel.base* reported the highest linking performance (SLM = 0.477) while *FBK-HLT-NLP* showed the best clustering (MC = 0.585) and tagging (STMM = 0.516) results. It is interesting to notice that all these systems (*UniPI*, *MicroNeel* and *FBK-HLT-NLP*) developed specific techniques for dealing with Twitter mentions reporting very good results for the tagging metric (with values always above 0.46).

All participants have made used of supervised algorithms at some point of their tagging/linking/clustering pipeline. *UniPi*, *Sisinflab* and *UNIMIB* have exploited word embeddings trained on the development set plus some

other external resources (manual annotated corpus, Wikipedia, and Twita). *UniPI* and *FBK-HLT-NLP* built additional training data obtained by active learning and manual annotation. The use of additional resources is allowed by the task guidelines, and both the teams have contributed to develop additional data useful for the research community.

7 Conclusions

We described the first evaluation task for entity linking in Italian tweets. The task evaluated the performance of participant systems in terms of (1) tagging entity mentions in the text of tweets; (2) linking the mentions with respect to the canonicalized DBpedia 2015-10; (3) clustering the entity mentions that refer to the same named entity.

The task has attracted many participants who specifically designed and developed algorithm for dealing with both Italian language and the specific peculiarity of text on Twitter. Indeed, many participants developed ad-hoc techniques for recognising Twitter mentions and hashtag. In addition, the participation in the task has fostered the building of new annotated datasets and corpora for the purpose of training learning algorithms and word embeddings.

We hope that this first initiative has set up the scene for further investigations and developments of best practises, corpora and resources for the Italian name entity linking on Tweets and other microblog contents.

As future work, we plan to build a bigger dataset of annotated contents and to foster the release of state-of-the-art methods for entity linking in Italian language.

Acknowledgments

This work is supported by the project "Multilingual Entity Liking" co-funded by the Apulia Region under the program FutureInResearch, by the ADAPT Centre for Digital Content Technology, which is funded under the Science Foundation Ireland Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund, and by H2020 FREME project (GA no. 644771).

References

Giuseppe Attardi, Daniele Sartiano, Maria Simi, and Irene Sucameli. 2016. Using Embeddings for Both Entity Recognition and Linking in Tweets. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Timothy Baldwin, Young-Bum Kim, Marie Catherine de Marneffe, Alan Ritter, Bo Han, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. *ACL-IJCNLP*, 126:2015.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta, Georgia, June. Association for Computational Linguistics.

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. 2015. Entity Linking for Italian Tweets. In Cristina Bosco, Sara Tonelli, and Fabio Massimo Zanzotto, editors, *Proceedings of the Second Italian Conference on Computational Linguistics CLiC-it 2015, Trento, Italy, December 3-8, 2015.*, pages 36–40. Accademia University Press.

Flavio Massimiliano Cecchini, Elisabetta Fersini, Enza Messina Pikakshi Manchanda, Debora Nozza, Matteo Palmonari, and Cezar Sas. 2016. UNIMIB@NEEL-IT : Named Entity Recognition and Linking of Italian Tweets. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Francesco Corcoglioniti, Alessio Palmero Aprosio, Yaroslav Nechaev, and Claudio Giuliano. 2016. MicroNeel: Combining NLP Tools to Perform Named Entity Detection and Linking on Microposts. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Vittoria Cozza, Wanda La Bruna, and Tommaso Di Noia. 2016. sisinflab: an ensemble of supervised

Table 3: Results of the evaluation with respect to: MC (*Mention_Ceaf*), STMM (*Strong_Typed_Mention_Match*), SLM (*Strong_Link_Match*) and the final score used for system ranking. Δ shows the final score improvement of the current system versus the previous. Best MC, STMM and SLM are reported in bold.

name	**MC**	**STMM**	**SLM**	**final score**	Δ
UniPI.3	0.561	0.474	0.456	0.5034	+1.27
UniPI.1	0.561	0.466	0.443	0.4971	+0.08
MicroNeel.base	0.530	0.472	**0.477**	0.4967	+0.10
UniPI.2	0.561	0.463	0.443	0.4962	+0.61
FBK-HLT-NLP.3	**0.585**	**0.516**	0.348	0.4932	+0.78
FBK-HLT-NLP.2	0.583	0.508	0.346	0.4894	+1.49
FBK-HLT-NLP.1	0.574	0.509	0.333	0.4822	+1.49
MicroNeel.merger	0.509	0.463	0.442	0.4751	+0.32
MicroNeel.all	0.506	0.460	0.444	0.4736	+38.56
sisinflab.1	0.358	0.282	0.380	0.3418	0.00
sisinflab.3	0.358	0.286	0.376	0.3418	+2.24
sisinflab.2	0.340	0.280	0.381	0.3343	+50.31
unimib.run_02	0.208	0.194	0.270	0.2224	+9.50
unimib.run_03	0.207	0.188	0.213	0.2031	+5.56
unimib.run_01	0.193	0.166	0.218	0.1924	0.00

and unsupervised strategies for the neel-it challenge at Evalita 2016. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Aba-Sah Dadzie, Daniel PreoÅčiuc-Pietro, Danica RadovanoviÄĞ, Amparo E. Cano Basave, and Katrin Weller, editors. 2016. *Proceedings of the 6th Workshop on Making Sense of Microposts*, volume 1691. CEUR.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 25–32. Association for Computational Linguistics.

Anne-Lyse Minard, R. H. Mohammed Qwaider, and Bernardo Magnini. 2016a. FBK-NLP at NEEL-IT: Active Learning for Domain Adaptation. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Anne-Lyse Minard, Manuela Speranza, and Tommaso Caselli. 2016b. The EVALITA 2016 Event Factuality Annotation Task (FactA). In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. 2016. Linking knowledge bases to social media profiles.

Alessio Palmero Aprosio and Claudio Giuliano. 2016. The Wiki Machine: an open source software for entity linking and enrichment. *ArXiv e-prints*.

Alessio Palmero Aprosio and Giovanni Moretti. 2016. Italy goes to Stanford: a collection of CoreNLP modules for Italian. *ArXiv e-prints*, September.

Giuseppe Rizzo, Marieke van Erp, Julien Plu, and Raphaël Troncy. 2016. Making Sense of Microposts (#Microposts2016) Named Entity rEcognition and Linking (NEEL) Challenge. In *6th Workshop on Making Sense of Microposts (#Microposts2016)*.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. Brat: A web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 102–107, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fabio Tamburini, Cristina Bosco, Alessandro Mazzei, and Andrea Bolioli. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing*

and Speech Tools for Italian (EVALITA 2016). aAcademia University Press.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! a free corpus-based morphological resource for the Italian language. *Corpus Linguistics 2005*, 1(1).

Using Embeddings for Both Entity Recognition and Linking in Tweets

Giuseppe Attardi, Daniele Sartiano, Maria Simi, Irene Sucameli
Dipartimento di Informatica
Università di Pisa
Largo B. Pontecorvo, 3
I-56127 Pisa, Italy
{attardi, sartiano, simi}@di.unipi.it
irenesucameli@gmail.com

Abstract

English. The paper describes our submissions to the task on Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) at Evalita 2016. Our approach relies on a technique of Named Entity tagging that exploits both character-level and word-level embeddings. Character-based embeddings allow learning the idiosyncrasies of the language used in tweets. Using a full-blown Named Entity tagger allows recognizing a wider range of entities than those well known by their presence in a Knowledge Base or gazetteer. Our submissions achieved first, second and fourth top official scores.

Italiano. *L'articolo descrive la nostra partecipazione al task di Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) a Evalita 2016. Il nostro approccio si basa sull'utilizzo di un Named Entity tagger che sfrutta embeddings sia character-level che word-level. I primi consentono di apprendere le idiosincrasie della scrittura nei tweet. L'uso di un tagger completo consente di riconoscere uno spettro più ampio di entità rispetto a quelle conosciute per la loro presenza in Knowledge Base o gazetteer. Le prove sottomesse hanno ottenuto il primo, secondo e quarto dei punteggi ufficiali.*

1 Introduction

Most approaches to entity linking in the current literature split the task into two equally important but distinct problems: *mention detection* is the task of extracting surface form candidates that correspond to entities in the domain of interest; *entity disambiguation* is the task of linking an extracted mention to a specific instance of an entity in a knowledge base.

Most approaches to mention detection rely on some sort of fuzzy matching between n-grams in the source text and a list of known entities (Rizzo et al., 2015). These solutions suffer severe limitations when dealing with Twitter posts, since the posts' vocabulary is quite varied, the writing is irregular, with variants and misspellings and entities are often not present in official resources like DBpedia or Wikipedia.

Detecting the correct entity mention is however crucial: Ritter et al. (2011) for example report a 0.67 F1 score on named entity segmentation, but an 85% accuracy, once the correct entity mention is detected, just by a trivial disambiguation that maps to the most popular entity.

We explored an innovative approach to mention detection, which relies on a technique of Named Entity tagging that exploits both character-level and word-level embeddings. Character-level embeddings allow learning the idiosyncrasies of the language used in tweets. Using a full-blown Named Entity tagger allows recognizing a wider range of entities than those well known by their presence in a Knowledge Base or gazetteer.

Another advantage of the approach is that no pre-built resource is required in order to perform the task, minimal preprocessing is required on the input text and no manual feature extraction nor feature engineering is required.

We exploit embeddings also for disambiguation and entity linking, proposing the first approach that, to the best of our knowledge, uses only embeddings for both entity recognition and linking.

We report the results of our experiments with this approach on the task Evalita 2016 NEEL-IT. Our submissions achieved first, second and fourth top official scores.

2 Task Description

The NEEL-IT task consists of annotating named entity mentions in tweets and disambiguating them by linking them to their corresponding entry in a knowledge base (DBpedia).

According to the task Annotation Guidelines (NEEL-IT Guidelines, 2016), a mention is a string in the tweet representing a proper noun or an acronym that represents an entity belonging to one of seven given categories (Thing, Event, Character, Location, Organization, Person and Product). Concepts that belong to one of the categories but miss from DBpedia are to be tagged as NIL. Moreover "The extent of an entity is the entire string representing the name, excluding the preceding definite article".

The Knowledge Base onto which to link entities is the Italian DBpedia 2015-10, however the concepts must be annotated with the canonicalized dataset of DBpedia 2015, which is an English one. Therefore, despite the tweets are in Italian, for unexplained reasons the links must refer to English entities.

3 Building a larger resource

The training set provided by the organizers consists of just 1629 tweets, which are insufficient for properly training a NER on the 7 given categories.

We thus decided to exploit also the training set of the Evalita 2016 PoSTWITA task, which consists of 6439 Italian tweets tokenized and gold annotated with PoS tags. This allowed us to concentrate on proper nouns and well defined entity boundaries in the manual annotation process of named entities.

We used the combination of these two sets to train a first version of the NER.

We then performed a sort of active learning step, applying the trained NER tagger to a set of over 10 thousands tweets and manually correcting 7100 of these by a team of two annotators.

These tweets were then added to the training set of the task and to the PoSTWITA annotated training set, obtaining our final training corpus of 13,945 tweets.

4 Description of the system

Our approach to Named Entity Extraction and Linking consists of the following steps:

- Train word embeddings on a large corpus of Italian tweets
- Train a bidirectional LSTM character-level Named Entity tagger, using the pre-trained word embeddings
- Build a dictionary mapping titles of the Italian DBpedia to pairs consisting of the corresponding title in the English DBpedia 2011 release and its NEEL-IT category. This helps translating the Italian titles into the requested English titles. An example of the entries in this dictionary are:

  Cristoforo_Colombo (http://dbpedia.org/resource/Christopher_Columbus, Person)
  Milano (http://dbpedia.org/resource/Milan, Location)

- From all the anchor texts from articles of the Italian Wikipedia, select those that link to a page that is present in the above dictionary. For example, this dictionary contains:

  Person Cristoforo_Colombo Colombo

- Create word embeddings from the Italian Wikipedia
- For each page whose title is present in the above dictionary, we extract its abstract and compute the average of the word embeddings of its tokens and store it into a table that associates it to the URL of the same dictionary
- Perform Named Entity tagging on the test set
- For each extracted entity, compute the average of the word embeddings for a context of words of size *c* before and after the entity.
- Annotate the mention with the DBpedia entity whose lc2 distance is smallest among those of the abstracts computed before.
- For the Twitter mentions, invoke the Twitter API to obtain the real name from the screen name, and set the category to Person if the real name is present in a gazetteer of names.

The last step is somewhat in contrast with the task guidelines (NEEL-IT Guidelines, 2016), which only contemplate annotating a Twitter mention as `Person` if it is recognizable on the spot as a known person name. More precisely "If the mention contains the name and surname of a person, the name of a place, or an event, etc., it

should be considered as a named entity", but "Should not be considered as named entity those aliases not universally recognizable or traceable back to a named entity, but should be tagged as entity those mentions that contains well known aliases. Then, `@ValeYellow46` should not be tagged as is not an alias for `Valentino Rossi`".

We decided instead that it would have been more useful, for practical uses of the tool, to produce a more general tagger, capable of detecting mentions recognizable not only from syntactic features. Since this has affected our final score, we will present a comparison with results obtained by skipping this last step.

4.1 Word Embeddings

The word embeddings for tweets have been created using the `fastText` utility[1] by Bojanowski et al. (2016) on a collection of 141 million Italian tweets retrieved over the period from May to September 2016 using the Twitter API. Selection of Italian tweets was achieved by using a query containing a list of the 200 most common Italian words.

The text of tweets was split into sentences and tokenized using the sentence splitter and the tweet tokenizer from the linguistic pipeline Tanl (Attardi et al., 2010), replacing emoticons and emojis with a symbolic name starting with EMO_ and normalizing URLs. This preprocessing was performed by MapReduce on the large source corpora.

We produced two versions of the embeddings, one with dimension 100 and a second of dimension 200. Both used a window of 5 and retained words with a minimum count of 100, for a total of 245 thousands words.

Word embeddings for the Italian Wikipedia were created from text extracted from the Wikipedia dump of August 2016, using the `WikiExtractor` utility by Attardi (2009). The vectors were produced by means of the `word2vec` utility[2], using the skipgram model, a dimension of 100, a window of 5, and a minimum occurrence of 50, retaining a total of 214,000 words.

4.2 Bi-LSTM Character-level NER

Lample et al. (2016) propose a Named Entity Recognizer that obtains state-of-the-art performance in NER on the 4 CoNLL 2003 datasets without resorting to any language-specific knowledge or resources such as gazetteers.

In order to take into account the fact that named entities often consist of multiple tokens, the algorithms exploits a bidirectional LSTM with a sequential conditional random layer above it.

Character-level features are learned while training, instead of hand-engineering prefix and suffix information about words. Learning character-level embeddings has the advantage of learning representations specific to the task and domain at hand. They have been found useful for morphologically rich languages and to handle the out-of-vocabulary problem for tasks like POS tagging and language modeling (Ling et al., 2015) or dependency parsing (Ballesteros et al., 2015).

The character-level embedding are given to bidirectional LSTMs and then concatenated with the embedding of the whole word to obtain the final word representation as described in Figure 1:

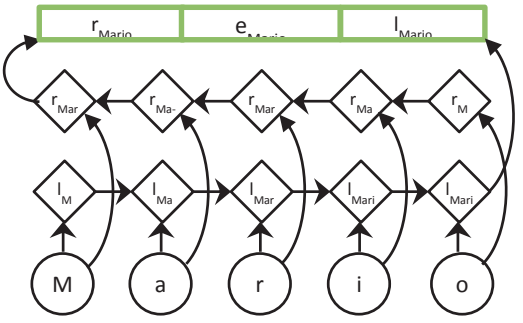


Figure 1. The embeddings for the word "Mario" are obtained by concatenating the two bidirectional LSTM character-level embeddings with the whole word embeddings.

The architecture of the NER tagger is described in Figure 2.

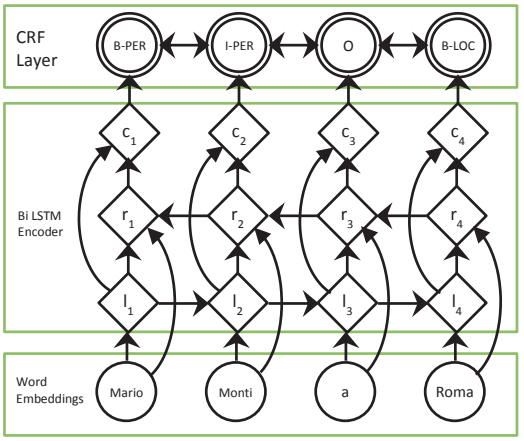


Figure 2. Architecture of the NER

[1] https://github.com/facebookresearch/fastText.git

[2] https://storage.googleapis.com/google-code-archive-source/v2/code.google.com/word2vec/source-archive.zip

5 Experiments

Since Named Entity tagging is the first step of our technique and hence its accuracy affects the overall results, we present separately the evaluation of the NER tagger.

Here are the results of the NER tagger on a development set of 1523 tweets, randomly extracted from the full training set.

Category	Precision	Recall	F1
Character	50.00	16.67	25.00
Event	92.48	87.45	89.89
Location	77.51	75.00	76.24
Organization	88.30	78.13	82.91
Person	73.71	88.26	88.33
Product	65.48	60.77	63.04
Thing	50.00	36.84	42.42

Table 1. NER accuracy on devel set.

On the subset of the test set used in the evaluation, which consists of 301 tweets, the NER performs as follows:

Category	Precision	Recall	F1
Character	0.00	0.00	0.00
Event	0.00	0.00	0.00
Location	72.73	61.54	66.67
Organization	63.46	44.59	52.38
Person	76.07	67.42	71.49
Product	32.26	27.78	29.85
Thing	0.00	0.00	0.00

Table 2. NER accuracy on gold test set.

In the disambiguation and linking process, we experimented with several values of the context size *c* of words around the mentions (4, 8 and 10) and eventually settled for a value of 8 in the submitted runs.

6 Results

We submitted three runs. The three runs have in common the following parameters for training the NER:

Character embeddings dimension	25
dropout	0.5
Learning rate	0.001
Training set size	12,188

Table 3. Training parameters for NER.

Specific parameters of the individual runs are:

- UniPI.1: twitter embeddings with dimension 100, disambiguation by frequency of mention in Wikipedia anchors
- UniPI.2: twitter embeddings with dimension 100, disambiguation with Wikipedia embeddings
- UniPI.3: twitter embeddings with dimension 200, disambiguation with Wikipedia embeddings, training set with geographical entities more properly annotated as Location (e.g. Italy).

The runs achieved the scores listed in the following table:

Run	Mention ceaf	Strong typed mention match	Strong link match	Final score
UniPI.3	0.561	0.474	0.456	**0.5034**
UniPI.1	0.561	0.466	0.443	0.4971
Team2.base	0.530	0.472	**0.477**	0.4967
UniPI.2	0.561	0.463	0.443	0.4962
Team3.3	**0.585**	**0.516**	0.348	0.4932

Table 4. Top official task results.

The final score is computed as follows:

$$0.4\ mention\_ceaf + 0.3\ strong\_typed\_mention\_match + 0.3\ strong\_link\_match$$

As mentioned, our tagger performs an extra effort in trying to determine whether Twitter mentions represent indeed Person or Organization entities. In order to check how this influences our result we evaluate also a version of the UniPI.3 run without the extra step of mention type identification. The results are reported in the following table:

Run	Mention ceaf	Strong typed mention match	Strong link match	Final score
UniPI.3 without mention check	**0.616**	0.531	0.451	**0.541**

Table 5. Results of run without Twitter mention analysis.

On the other hand, if we manually correct the test set annotating the Twitter mentions that indeed refer to Twitter users or organizations, the score for strong typed mentions match increases to 0.645**.**

7 Discussion

The effectiveness of the use of embeddings in disambiguation can be seen in the improvement in the strong link match score between run UniPI.2 and UniPI.3. Examples where embeddings lead to better disambiguation are:

Liverpool_F.C. vs Liverpool
Italy_national_football_team vs Italy
S.S._Lazio vs Lazio
Diego_Della_Valle vs Pietro_Della_Valle
Nobel_Prize vs Alfred_Nobel

There are many cases where the NER recognizes a Person, but the linker associates the name to a famous character, for example:

Maria_II_of_Portugal for Maria
Luke_the_Evangelist for Luca

The approach of using embeddings for disambiguation looks promising: the abstract of articles sometimes does not provide appropriate evidence, since the style of Wikipedia involves providing typically meta-level information, such as the category of the concept. For example disambiguation for "Coppa Italia" leads to "Italian_Basketball_Cup" rather than to "Italian_Football_Cup", since both are described as sport competitions. Selecting or collecting phrases that mention the concept, rather than define it, might lead to improved accuracy.

Using character-based embeddings and a large training corpus requires significant computational resources. We exploited a server equipped with nVidia Tesla K 80 GPU accelerators.

Nevertheless training the LSTM NER tagger still required about 19 hours: without the GPU accelerator the training would have been impossible.

8 Related Work

Several papers discuss approaches to end-to-end entity linking (Cucerzan, 2007; Milne and Witten, 2008; Kulkarni et al., 2009; Ferragina and Scaiella, 2010; Han and Sun, 2011; Meij et al., 2012), but many heavily depend on Wikipedia text and might not work well in short and noisy tweets.

Most approaches to mention detection rely on some sort of fuzzy matching between n-grams in the source and the list of known entities (Rizzo et al., 2015).

Yamada et al. (2015) propose an end-to-end approach to entity linking that exploits word embeddings as features in a random-forest algorithm used for assigning a score to mention candidates, which are however identified by either exact or fuzzy matching on a mention-entity dictionary built from Wikipedia titles and anchor texts.

Guo et al. (2016) propose a structural SVM algorithm for entity linking that jointly optimizes mention detection and entity disambiguation as a single end-to-end task.

9 Conclusions

We presented an innovative approach to mention extraction and entity linking of tweets, that relies on Deep Learning techniques. In particular we use a Named Entity tagger for mention detection that exploits character-level embeddings in order to deal with the noise in the writing of tweet posts. We also exploit word embeddings as a measure of semantic relatedness for the task of entity linking.

As a side product we produced a new gold resource of 13,609 tweets (242,453 tokens) annotated with NE categories, leveraging on the resource distributed for the Evalita PoSTWITA task.

The approach achieved top score in the Evalita 2016 NEEL-IT Challenge and looks promising for further future enhancements.

Acknowledgments

We gratefully acknowledge the support by the University of Pisa through project PRA 2016 and by NVIDIA Corporation through the donation of a Tesla K40 GPU accelerator used in the experiments.

We thank Guillaume Lample for making available his implementation of Named Entity Recognizer.

References

Giuseppe Attardi. 2009. WikiExtractor: A tool for extracting plain text from Wikipedia dumps. https://github.com/attardi/wikiextractor

Giuseppe Attardi, Stefano Dei Rossi, and Maria Simi. 2010. The Tanl Pipeline. In *Proc. of LREC Workshop on WSPP*, Malta.

Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2015. Improved transition-based dependency parsing by modeling characters instead of words with LSTMs. In *Proceedings of EMNLP 2015*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vec-

tors with Subword Information. https://arxiv.org/abs/1607.04606

Silviu Cucerzan. 2007. Large-scale named entity disam- biguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference of EMNLP-CoNLL*, pages 708–716.

Evalita. 2016. NEEL-IT. http://neel-it.github.io

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: on-the-fly annotation of short text fragments (by wikipedia enti- ties). In *Proceedings of the 19th ACM international conference on Information and knowledge manage- ment*, CIKM '10, pages 1625–1628, New York, NY, USA. ACM.

Stephen Guo, Ming-Wei Chang and Emre Kıcıman. 2016. To Link or Not to Link? A Study on End-to-End Tweet Entity Linking.

Xianpei Han and Le Sun. 2011. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of the 49th Annual Meeting of the Asso- ciation for Computational Linguistics: Human Lan- guage Technologies - Volume 1*, HLT '11, pages 945– 954, Stroudsburg, PA, USA. Association for Compu- tational Linguistics.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD), pages 457–466, New York, NY, USA. ACM.

Guillaume Lample, Miguel Ballesteros, Kazuya Kawakami, Sandeep Subramanian, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition, In *Proceedings of NAACL-HLT (NAACL 2016).*

Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. 2012. Adding semantics to microblog posts. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 563–572, New York, NY, USA. ACM.

David Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *Proceedings of ACM Conference on Information and Knowledge Management (CIKM)*, pages 509–518, New York, NY, USA. ACM.

NEEL-IT Annotation Guidelines. 2016. https://drive.google.com/open?id=1saUb2NSxml67pcrz3m_bMcibe1nST2CTedeKOBklKaI

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*. arXiv:1310.4546

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 1524–1534, Stroudsburg, PA, USA. Association for Computational Linguistics.

Giuseppe Rizzo, Amparo Elizabeth Cano Basave, Bianca Pereira, and Andrea Varga. 2015. Making sense of microposts (#microposts2015) named entity recognition and linking (NEEL) challenge. In *Proceedings of the 5th Workshop on Making Sense of Microposts*, pages 44–53.

Ikuya Yamada, Hideaki Takeda, and Yoshiyasu TakefujiIkuya. 2015. An End-to-End Entity Linking Approach for Tweets. In *Proceedings of the 5th Workshop on Making Sense of Microposts*, Firenze, Italy.

UNIMIB@NEEL-IT : Named Entity Recognition and Linking of Italian Tweets

Flavio Massimiliano Cecchini, Elisabetta Fersini, Pikakshi Manchanda, Enza Messina, Debora Nozza, Matteo Palmonari, Cezar Sas
Department of Informatics, Systems and Communication (DISCo)
University of Milano-Bicocca, Milan, Italy
{flavio.cecchini, fersini, pikakshi.manchanda, messina, debora.nozza, palmonari}@disco.unimib.it
c.sas@campus.unimib.it

Abstract

English. This paper describes the framework proposed by the UNIMIB Team for the task of Named Entity Recognition and Linking of Italian tweets (NEEL-IT). The proposed pipeline, which represents an entry level system, is composed of three main steps: (1) Named Entity Recognition using Conditional Random Fields, (2) Named Entity Linking by considering both Supervised and Neural-Network Language models, and (3) NIL clustering by using a graph-based approach.

Italiano.

Questo articolo descrive il sistema proposto dal gruppo UNIMIB per il task di Named Entity Recognition and Linking applicato a tweet in lingua italiana (NEEL-IT). Il sistema, che rappresenta un approccio iniziale al problema, è costituito da tre passaggi fondamentali: (1) Named Entity Recognition tramite l'utilizzo di Conditional Random Fields, (2) Named Entity Linking considerando sia approcci supervisionati sia modelli di linguaggio basati su reti neurali, e (3) NIL clustering tramite un approccio basato su grafi.

1 Introduction

Named Entity Recognition (NER) and Linking (NEL) have gained significant attention over the last years. While dealing with short textual formats, researchers face difficulties in such tasks due to the increasing use of informal, concise and idiosyncratic language expressions (Derczynski et al., 2015). In this paper, we introduce a system that tackles the aforementioned issues for **Italian language** tweets. A detailed description of these tasks is provided in the next sections.

2 Systems Description

The proposed system (Figure 1) comprises of three stages: Named Entity Recognition, Named Entity Linking and NIL Clustering. In this section,

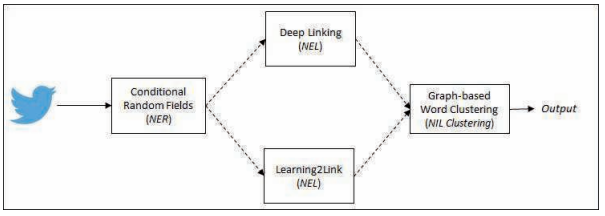


Figure 1: UNIMIB system. Dotted paths are related to optional paths.

we provide a detailed explanation of the different methods used to address these tasks.

2.1 Named Entity Recognition

In order to identify named entities from microblog text, we used Conditional Random Fields (CRF), i.e. a probabilistic undirected graphical model that defines the joint distribution $P(y|x)$ of the predicted labels (hidden states) $y = y_1, ..., y_n$ given the corresponding tokens (observations) $x = x_1, ..., x_n$. The probability of a sequence of label y given the sequence of observations x can be rewritten as:

$$p(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{t=1}^{N} \sum_{k=1}^{K} \omega_k f_k(y_t, y_{t-1} x, t)\right) \quad (1)$$

where $f_k(y_t, y_{t-1}x, t)$ is an arbitrary feature function over its arguments and ω_k is a feature weight that is a free parameter in the model. Feature

functions are fixed in advance and are used to verify some properties of the input text, while the weights ω_k have to be learned from data and are used to tune the discriminative power of each feature function. In our runs, two configurations of CRF have been trained using the training data available for the challenge: (1) CRF and (2) CRF+Gazetteers. In particular, in the last configuration the model has been induced enclosing several gazetteers, i.e. products, organizations, persons, events and characters. The output of CRF is a set candidate entities $e_1, e_2, ..., e_m$ in each given tweet t.

2.2 Named Entity Linking

The task of Named Entity Linking (NEL) is defined as associating an entity mention e_j (identified from a tweet t) to an appropriate KB candidate resource c_j^i from a set $C_j = \{c_j^1, c_j^2, \cdots, c_j^k\}$ of candidate resources. We explored two different linking approaches: Learning2Link and Neural-Network Language Model (NNLM) Linking.

2.2.1 Learning2Link

For this phase, we used the Italian version of DBpedia as our KB. To this end, we extract Titles of all Wikipedia articles (i.e., the *labels* dataset) from Italian DBpedia and index them using LuceneAPI. For each entity mention e_j, we retrieve a list of top-k ($k = 10$) candidate resources from the KB. We compute the scores as described below (Caliano et al., 2016), which are used to create the input space for the Learning2Link (L2L) phase for each candidate resource for an entity mention:

- $lcs(e_j, l_{c_j^i})$ which denotes a normalized Lucene Conceptual Score between an entity e_j and the label of a candidate resource $l_{c_j^i}$;
- $cos(e_j^*, a_{c_j^i}))$ which represents a discounted cosine similarity between an entity context e_j^* (modeled as a vector composed of an identified entity e_j and non stop-words in a tweet t) and a candidate KB abstract description $a_{c_j^i}$;
- *Jaro-Winkler distance* (Jaro, 1995) between an entity e_j and the label of a resource $l_{c_j^i}$;
- $R(c_j^i)$ which is a popularity measure of a given candidate resource c_j^i in the KB.

This input space is used for training various learning algorithms such as *Decision Trees (DT), Multi-Layer Perceptron (MLP), Support Vector Machines (SVM)* with Linear-, Polynomial- and Radial-kernels, *Bayesian Networks (BN), Voted Perceptron (VP), Logistic Regression (LR)* and *Naïve Bayes* (NB). The target class is a boolean variable which indicates whether or not a candidate resource URI is a suitable link in the KB for the entity mention e_j. An important point to note here is that the models are learning by similarity, i.e., they learn the target class for a candidate resource by using the afore-mentioned similarity scores.

A **Decision Criteria** is further created based on the target class so as to predict the most suitable candidate resource URI from amongst a list of URIs of candidate resources $\{c_j^1, c_j^2, \cdots, c_j^k\}$ of an entity mention e_j (or detect the NIL mentions) in the test set. This criteria is described as follows:

if candidate resource c_j^i is predicted to be a suitable match for e_j **then**
  Map the entity mention e_j to the candidate resource c_j^i
else if more than one candidate resources have been predicted to be suitable matches for e_j **then**
  Map the entity mention e_j to the candidate resource c_j^i with the highest probability score
else if no candidate resource is predicted as a suitable match by the algorithm **then**
  Map the entity mention e_j to a NIL mention
end if

Finally, the entity type of a mention is determined by the DBpedia type of the selected candidate resource, which is finally mapped to a type in the Evalita Ontology based on an Ontology mapping that we developed between the Evalita Ontology and the DBpedia Ontology, as per the guidelines of the Challenge. In case, a mention has been mapped to a NIL mention, the entity type is determined by the CRF type obtained in the entity recognition phase.

2.2.2 Neural-Network Language Model (NNLM) Linking

The process of generating the candidate resource set C_j for the entity mention e_j is a crucial part for the NEL task. To obtain C_j, most of the state-of-the-art approaches (Dredze et al., 2010; McNamee, 2010) make use of exact or partial matching (e.g. Hamming distance, character Dice score, etc.) between the entity mention e_j and the labels

of all the resources in the KB. However, these approaches can be error-prone, especially when dealing with microblog posts rich of misspellings, abbreviations, nicknames and other noisy forms of text.

The idea behind the proposed NNLM Linking approach is to exploit a high-level similarity measure between the entity mentions e_j and the KB resources, in order to deal with the afore-mentioned issues. Instead of focusing on the similarity measure definition, we focus on the word representation. The need of a meaningful and dense representation of words, where words and entities are represented in a different way, and an efficient algorithm to compute this representation, lead us to the most used Neural-Network Language model, i.e. Word Embeddings (Mikolov et al., 2013).

A Word Embedding, $WE : words \rightarrow \mathbb{R}^n$, is a function which maps words in some language to high-dimensional vectors. Embeddings have been trained on the Italian Wikipedia and they have been generated for all the words in the Wikipedia texts, adding a specific tag if the words corresponded to a KB entry, i.e. a Wikipedia article.

Given an entity e_j and a word w belonging to the word's dictionary D of the Wikipedia text, we can define the similarity function s as:

$$s(e_j, w) = sim(\, WE(e_j),\, WE(w)), \qquad (2)$$

where sim is the cosine similarity.

Given an entity e_j, the candidate resource set C_j is created by taking the top-k words w for the similarity score $s(e_j, w)$. Then, the predicted resource c^* is related to the word with the highest similarity score such that the word corresponds to a KB entry and its type is coherent with the type resulting from the NER system. If C_j does not contain words correspondent to a KB entry, e_j is considered as a NIL entity.

2.3 NIL Clustering

We tackled the subtask of NIL clustering with a graph-based approach. We build a weighted, undirected co-occurrence graph where an edge represents the co-occurrence of two terms in a tweet. Edge weights are the frequencies of such co-occurrences. We did not use measures such as log likelihood ratio or mutual information, as frequencies might be too low to yield significant scores. In the word graph we just retained lemmatized nouns, verbs, adjectives and proper nouns, along with abbreviations and foreign words. More precisely, we used TreeTagger (Schmid, 1994) with Achim Stein's parameters for Italian part-of-speech tagging, keeping only tokens tagged as `VER`, `NOM`, `NPR`, `ADJ`, `ABR`, `FW` and `LS`. We made the tagger treat multi-word named entities (be they linked or NIL) as single tokens. The ensuing word graph was then clustered using the MaxMax algorithm (Hope and Keller, 2013) to separate it into rough topical clusters. We notice that tweets with no words in common always lie in different connected components of the word graph and thus in different clusters.

Subsequently, we reduced the clusters considering only tokens that were classified as NILs. Within each cluster, we measure the string overlap between each pair of NIL tokens s_1, s_2, assigning it a score in $[0, 1]$. We computed the length λ of the longest prefix[1] of the shorter string that is also contained in the longer string and assigned it the score $\frac{\lambda^2}{|s_1| \cdot |s_2|}$. Similar overlaps of two or less letters, i.e. when $\lambda \leq 2$, are not considered meaningful, so they automatically receive a score of 0; on the contrary, when two meaningfully long strings coincide, i.e. $\lambda = |s_1| = |s_2|$ and $|s_1| > 2$, the pair will receive a score of 1.

A token is considered to possibly represent the same entity as another token if 1) their named entity type is the same and 2a) their overlap score is greater than an experimentally determined threshold) or 2b) they co-occur in any tweet and their overlap score is greater than 0. For each token s, we consider the set of other tokens that satisfy 1) and 2a) or 2b) for s. However, this still does not define an equivalence relation, so that we have to perform intersection and union operations on these sets to obtain the final partition of the NIL tokens. Finally, each NIL named entity will be labelled according to its cluster.

3 Results and Discussion

We first evaluate our approach on the training set consisting of 1000 tweets made available by the EVALITA 2016 NEEL-IT challenge. The results have been obtained by performing a 10-folds cross-validation. For each stage, we report the performance measures computed independently from the precedent phases.

[1] A prefix of length n is defined here as the first n letters of a string.

In the last subsection we report the results obtained on the test set for the three run submitted:

- *run 01*: *CRF* as NER approach and *NNLM Linking* as NEL system;
- *run 02*: *CRF+Gazetteers* as NER approach and *NNLM Linking* as NEL system;
- *run 03*: *CRF+Gazetteers* as NER approach and *Learning2Link* with Decision Tree (DT) as NEL system.

3.1 Named Entity Recognition

We report the results of CRF, in terms of Precision (P), Recall (R) and F1-Measure (F1) in Table 1, according to the two investigated configurations: CRF and CRF+Gazetteers. First of all, we can note the poor recognition performances obtained in both configurations, which are mainly due to the limited amount of training data. These poor performances are highlighted even more by looking at the entity types *Thing* (20), *Event* (15) and *Character* (18), whose limited number of instances do not allow CRF to learn any linguistic pattern to recognize them. For the remaining types, CRF+Gazetteers is able to improve Precision but at some expenses of Recall.

Table 1: Entity Recognition Results

	CRF			**CRF+Gazetteers**		
Label	P	R	F1	P	R	F1
Thing	0	0	0	0	0	0
Event	0	0	0	0	0	0
Character	0	0	0	0	0	0
Location	0.56	0.40	0.47	0.64	0.40	0.5
Organization	0.43	0.24	0.31	0.60	0.20	0.30
Person	0.50	0.30	0.37	0.69	0.21	0.33
Product	0.12	0.11	0.11	0.31	0.10	0.16
Overall	0.37	0.24	0.29	0.57	0.20	0.30

The low recognition performance have a great impact on the subsequent steps of the pipeline. To this purpose, we will report the result of Entity Linking and NIL clustering by considering an oracle NER (i.e. a perfect named entity recognition system) in the following subsections.

3.2 Named Entity Linking

We report the Precision (P), Recall (R) and F-measure (F1) of the Strong Link Match (SLM) measure for each addressed approach for NEL in Table 2. The results have been computed assuming the NER system as an oracle, i.e., every entity mention is correctly recognized and classified.

Table 2: Strong Link Match measure.

	P	**R**	**F1**
NNLM Linking	0.619	**0.635**	**0.627**
L2L DT	0.733	0.371	0.492
L2L MLP	0.684	0.333	0.448
L2L NB	0.614	0.312	0.414
L2L LR	0.709	0.278	0.399
L2L SVM-Polynomial	0.721	0.27	0.393
L2L VP	0.696	0.274	0.393
L2L BN	**0.741**	0.266	0.392
L2L SVM-Radial	0.724	0.264	0.387
L2L SVM-Linear	0.686	0.266	0.384

Regarding the Learning2Link approach, we evaluate the results for each machine learning model considered. Although the low performances in terms of F-measure, we can highlight that Decision Tree (DT) is a leaner algorithm with the highest Strong Link Match F-measure. On the other hand, low recall scores could be attributed to the inability of the retrieval system to find the "correct" link in the top-10 candidate list. A list of irrelevant candidate resources results in uninformative similarity scores, which causes the learning models to predict a target class where none of the candidate resources is a suitable match for an entity mention.

NNLM Linking shows significant results, proving the importance of not considering an entity mention as a mere string but instead use a representation that is able to capture a deeper meaning of the word/entity.

3.3 NIL Clustering

Assuming every non-NIL entity has been correctly classified, our system for NIL clustering achieves a CEAF score of 0.994. We remark that NILs in the data set are very fragmented and a baseline system of one cluster per entity is capable of reaching a score of 0.975. Our algorithm however puts NILs represented in the tweets by the same string or sharing a significant portion of their strings in the same cluster; the reason why it does not get a perfect score is that either the same entity appears in tweets not sharing common words, and thus belonging to different components of the word graph (same NIL, different clusters), or that two entities are too similar and there is not enough context to distinguish them (different NILs, same cluster). As the data set is very sparse, these phenomena are

Table 3: Experimental results on the test set

run ID	MC	STMM	SLM	Score
run 01	0.193	0.166	0.218	0.192
run 02	**0.208**	**0.194**	**0.270**	**0.222**
run 03	0.207	0.188	0.213	0.203

very likely to occur. Finally, we notice that the NIL clustering performance strongly depends on the Named Entity Recognition and Linking output: if two occurrences of the same NIL are mistakenly assigned to different types, they will never end up in the same cluster.

3.4 Overall

The results of the submitted runs are reported in Table 3. The first column shows the given configuration, the other columns report respectively the F-measure of: Strong Link Match (SLM), Strong Typed Mention Match (STMM) and Mention Ceaf (MC).

As a first consideration we can highlight that involving CRF (*run 01*), instead of the CRF+Gazetteers configuration (*run 02* and *run 03*), has lead to a significant decrease of the performance, even more substantial than the one reported in Section 3.1.

Given the best NER configuration, the NNLM approach (*run 02*) is the one with better performances confirming the results presented in Section 3.2. As expected, the low recognition performance of the NER system strongly affected the NEL performance resulting in low results compared to the ones obtained considering an oracle NER.

The main limitation of the proposed pipeline emerged to be the Named Entity Recognition step. As mentioned before, one of the main problems is the availability of training data to induce the probabilistic model. A higher number of instances could improve the generalization abilities of Conditional Random Fields, resulting in a more reliable named entity recognizer. An additional improvement concerns the inclusion of information related to the Part-Of-Speech in the learning (and inference) phase of Conditional Random Fields. To this purpose, the Italian TreeTagger could be adopted to obtain the Part-Of-Speech for each token in tweets and to enclose this information into the feature functions of Conditional Random Fields. A further improvement relates to the use of extended gazetteers (not only related to the Italian language) especially related to the types *Event* and *Character* (which in most of the cases are English-based named entities). A final improvement could be achieved by introducing an additional step between the named entity recognition and the subsequent steps. To this purpose, the available Knowledge Base could be exploited as distant supervision to learn a "constrained" Topic Model (Blei et al., 2003) able to correct the type prediction given by Conditional Random Fields. This solution could not only help to overcome the limitation related to the reduced number of training instances, but could also have a good impact in terms of type corrections of named entities.

4 Conclusion

In this paper, we described a Named Entity Recognition and Linking framework for microposts that participated in EVALITA 2016 NEEL-IT challenge as UNIMIB team. We further provided an overview of our system for recognizing entity mentions from Italian tweets and introduced novel approach for linking them to suitable resources in an Italian knowledge base.

We observed a particularly poor performance of the Coditional Random Fields in the Named Entity Recognition phase, mainly due to lack of appropriate instances of entity types. Regarding the Named Entity Linking step, NNLM Linking shows significant results, proving the importance of a high-level representation able to capture deeper meanings of entities. Further, the Learning2Link phase turns out to be a promising approach, given the small amount of training instances, although, there is a considerable scope for improvement if more candidate resources are used. Other similarity measures can also be experimented with, while studying their impact on the feature space.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Davide Caliano, Elisabetta Fersini, Pikakshi Manchanda, Matteo Palmonari, and Enza Messina. 2016. Unimib: Entity linking in tweets using jaro-winkler distance, popularity and coherence. In *Proceedings of the 6th International Workshop on Making Sense of Microposts (# Microposts)*.

Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke van Erp, Genevieve Gorrell, Raphaël

Troncy, Johann Petrak, and Kalina Bontcheva. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 277–285. Association for Computational Linguistics.

David Hope and Bill Keller. 2013. Maxmax: a graph-based soft clustering algorithm applied to word sense induction. In *Computational Linguistics and Intelligent Text Processing*, pages 368–381. Springer.

Matthew A Jaro. 1995. Probabilistic linkage of large public health data files. *Statistics in medicine*, 14(5-7):491–498.

Paul McNamee. 2010. Hltcoe efforts in entity linking at tac kbp 2010. In *Proceedings of the 3rd Text Analysis Conference Workshop*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3:1–12, jan.

Helmut Schmid. 1994. Probabilistic part-of speech tagging using decision trees. In *New methods in language processing*, page 154. Routledge.

MicroNeel: Combining NLP Tools to Perform Named Entity Detection and Linking on Microposts

Francesco Corcoglioniti, Alessio Palmero Aprosio, Yaroslav Nechaev, Claudio Giuliano
Fondazione Bruno Kessler
Trento, Italy
{corcoglio,aprosio,nechaev,giuliano}@fbk.eu

Abstract

English. In this paper we present the MicroNeel system for Named Entity Recognition and Entity Linking on Italian microposts, which participated in the NEEL-IT task at EVALITA 2016. MicroNeel combines The Wiki Machine and Tint, two standard NLP tools, with comprehensive tweet preprocessing, the Twitter-DBpedia alignments from the Social Media Toolkit resource, and rule-based or supervised merging of produced annotations.

Italiano. *In questo articolo presentiamo il sistema MicroNeel per il riconoscimento e la disambiguazione di entità in micropost in lingua Italiana, con cui abbiamo partecipato al task NEEL-IT di EVALITA 2016. MicroNeel combina The Wiki Machine e Tint, due sistemi NLP standard, con un preprocessing esteso dei tweet, con gli allineamenti tra Twitter e DBpedia della risorsa Social Media Toolkit, e con un sistema di fusione delle annotazioni prodotte basato su regole o supervisionato.*

1 Introduction

Microposts, i.e., brief user-generated texts like tweets, checkins, status messages, etc., are a form of content highly popular on social media and an increasingly relevant source for information extraction. The application of Natural Language Processing (NLP) techniques to microposts presents unique challenges due to their informal nature, noisiness, lack of sufficient textual context (e.g., for disambiguation), and use of specific abbreviations and conventions like #hashtags, @user mentions, retweet markers and so on. As a consequence, standard NLP tools designed and trained on more 'traditional' formal domains, like news article, perform poorly when applied to microposts and are outperformed by NLP solutions specifically-developed for this kind of content (see, e.g., Bontcheva et al. (2013)).

Recognizing these challenges and following similar initiatives for the English language, the NEEL-IT[1] task (Basile et al., 2016a) at EVALITA 2016[2] (Basile et al., 2016b) aims at promoting the research on NLP for the analysis of microposts in the Italian language. The task is a combination of Named Entity Recognition (NER), Entity Linking (EL), and Coreference Resolution for Twitter tweets, which are short microposts of maximum 140 characters that may include hashtags, user mentions, and URLs linking to external Web resources. Participating systems have to recognize mentions of named entities, assign them a NER category (e.g., person), and disambiguate them against a fragment of DBpedia containing the entities common to the Italian and English DBpedia chapters; unlinked (i.e., NIL) mentions have finally to be clustered in coreference sets.

In this paper we present our MicroNeel system that participated in the NEEL-IT task. With MicroNeel, we investigate the use on microposts of two standard NER and EL tools – The Wiki Machine (Palmero Aprosio and Giuliano, 2016) and Tint (Palmero Aprosio and Moretti, 2016) – that were originally developed for more formal texts. To achieve adequate performances, we complement them with: (i) a preprocessing step where tweets are enriched with semantically related text, and rewritten to make them less noisy; (ii) a set of alignments from Twitter user mentions to DBpedia entities, provided by the Social Media Toolkit (SMT) resource (Nechaev et al., 2016); and (iii) rule-based and supervised mechanisms for merging the annotations produced by NER, EL, and SMT, resolving possible conflicts.

[1] http://neel-it.github.io/
[2] http://www.evalita.it/2016

In the remainder of the paper, Section 2 introduces the main tools and resources we used. Section 3 describes MicroNeel, whose results at NEEL-IT and their discussions are reported in Sections 4 and 5. Section 6 presents the system open-source release, while Section 7 concludes.

2 Tools and Resources

MicroNeel makes use of a certain number of resources and tools. In this section, we briefly present the main ones used in the annotation process. The description of the rest of them (mainly used for preprocessing) can be found in Section 3.

2.1 The Wiki Machine

The Wiki Machine[3] (Palmero Aprosio and Giuliano, 2016) is an open source Entity Linking tool that automatically annotates a text with respect to Wikipedia pages. The output is provided through two main steps: entity identification, and disambiguation. The Wiki Machine is trained using data extracted from Wikipedia and is enriched with Airpedia (Palmero Aprosio et al., 2013), a dataset built on top of DBpedia (Lehmann et al., 2015) that increase its coverage over Wikipedia pages.

2.2 Tint

Tint[4] (Palmero Aprosio and Moretti, 2016) is an easy-to-use set of fast, accurate and extensible Natural Language Processing modules for Italian. It is based on Stanford CoreNLP[5] and is distributed open source. Among other modules, the Tint pipeline includes tokenization, sentence splitting, part-of-speech tagging and NER.

2.3 Social Media Toolkit

Social Media Toolkit[6] (Nechaev et al., 2016), or SMT, is an API that is able to align any given knowledge base entry to a corresponding social media profile (if it exists). The reverse alignment is achieved by using a large database (~1 million entries) of precomputed alignments between DBpedia and Twitter. SMT is also able to classify any Twitter profile as a person, organization, or other.

3 Description of the System

MicroNeel accepts a micropost text as input, which may include hashtags, mentions of Twitter users, and URLs. Alternatively, a tweet ID can be supplied in input (as done in NEEL-IT), and the system retrieves the corresponding text and metadata (e.g., author information, date and time, language) from Twitter API, if the tweet has not been deleted by the user or by Twitter itself.

Processing in MicroNeel is structured as a pipeline of three main steps, outlined in Figure 1: *preprocessing*, *annotation*, and *merging*. Their execution on an example tweet is shown in Figure 2.

3.1 Preprocessing

During the first step, the *original text* of the micropost is rewritten, keeping track of the mappings between original and rewritten offsets. The *rewritten text* is obtained by applying the following transformations:

- Hashtags in the text are replaced with their tokenizations. Given an hashtag, a bunch of 100 tweets using it is retrieved from Twitter. Then, when some camel-case versions of that hashtag are found, tokenization is done based on the sequence of uppercase letters used.
- User mentions are also replaced with their tokenizations (based on camel-case) or the corresponding display names, if available.
- Slangs, abbreviations, and some common typos (e.g., e' instead of è) in the text are replaced based on a custom dictionary (for Italian, we extracted it from the Wikipedia page `Gergo_di_Internet`[7]).
- URLs, emoticons, and other unprocessable sequences of characters in the text are discarded.
- True-casing is performed to recover the proper word case where this information is lost (e.g., all upper case or lower case text). This task employs a dictionary, which for Italian is derived from Morph-It! (Zanchetta and Baroni, 2005).

To help disambiguation, the rewritten text is then augmented with a textual *context* obtained by aggregating the following contents, if available:

- Hashtag descriptions from *tagdef*[8], a collaborative online service;

[3] `http://thewikimachine.fbk.eu/`
[4] `http://tint.fbk.eu/`
[5] `http://stanfordnlp.github.io/CoreNLP/`
[6] `http://alignments.futuro.media/`
[7] `https://it.wikipedia.org/wiki/Gergo_di_Internet`
[8] `https://www.tagdef.com/`

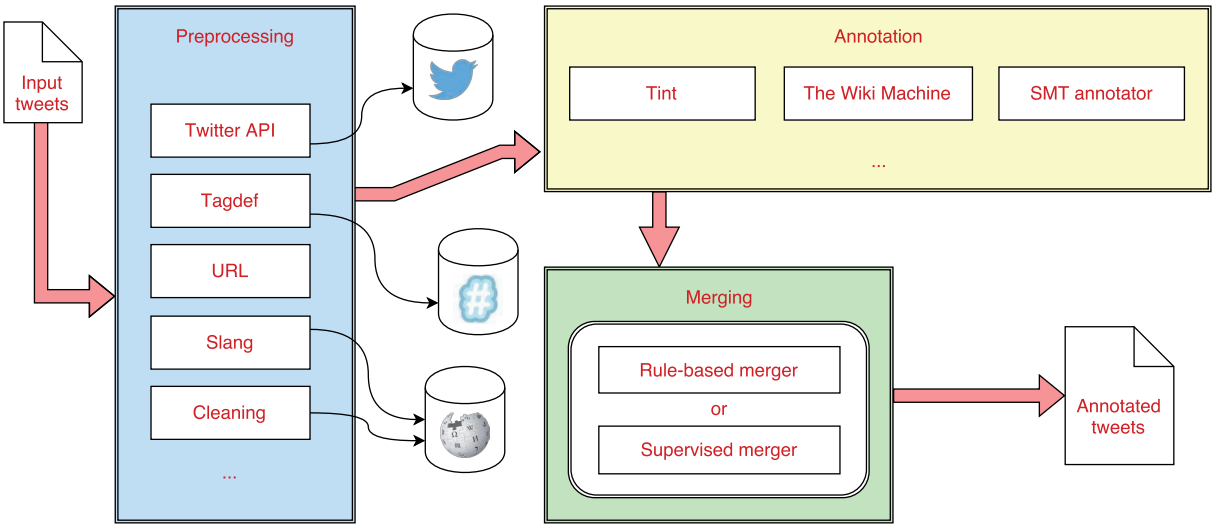


Figure 1: The overview of the system.

- Twitter user descriptions for author and user mentions in the original text;
- Titles of web pages linked by URLs in the original text.

In the example shown in Figure 2, from the original tweet

[Original text]
(author: @OscardiMontigny)
#LinkedIn: 200 milioni di iscritti, 4 milioni in Italia http://t.co/jK8MRiaS via @vincos

we collect

- metadata information for the author (Twitter user @OscardiMontigny);
- description of the hashtag #LinkedIn;
- title of the URL http://t.co/jK8MRiaS;
- metadata information for the Twitter user @vincos, mentioned in the tweet.

The resulting (cleaned) tweet is

[Rewritten text]
LinkedIn: 200 milioni di iscritti, 4 milioni in Italia via Vincenzo Cosenza

with context

[Context]
Speaker; Blogger; Mega-Trends, Marketing and Innovation Divulgator. #linkedin is about all things from Linkedin. LinkedIn: 200 milioni di iscritti, 4 milioni in Italia — Vincos Blog. Strategist at @BlogMeter My books: Social Media ROI — La società dei dati.

3.2 Annotation

In the second step, annotation is performed by three independent annotator tools run in parallel:

- The rewritten text is parsed with the NER module of Tint (see Section 2.2). This processing annotates named entities of type person, organization, and location.

- The rewritten text, concatenated with the context, is annotated by The Wiki Machine (see Section 2.1) with a list of entities from the full Italian DBpedia. The obtained EL annotations are enriched with the DBpedia class (extended with Airpedia), and mapped to the considered NER categories (person, organization, location, product, event).

- The user mentions in the tweet are assigned a type and are linked to the corresponding DBpedia entities using SMT (see Section 2.3); as for the previous case, SMT types and DBpedia classes are mapped to NER categories. A problem here is that many user mentions classified as persons or organizations by SMT are non-annotable according to NEEL-IT guidelines.[9] Therefore, we implemented two strategies for deciding whether to annotate a user mention:

[9]Basically, a user mention can be annotated in NEEL-IT if its NER category can be determined by just looking at the username and its surrounding textual context in the tweet. Usernames resembling a person or an organization name are thus annotated, while less informative usernames are not marked as their nature cannot be determined without looking at their Twitter profiles or at the tweets they made, which is done instead by SMT.

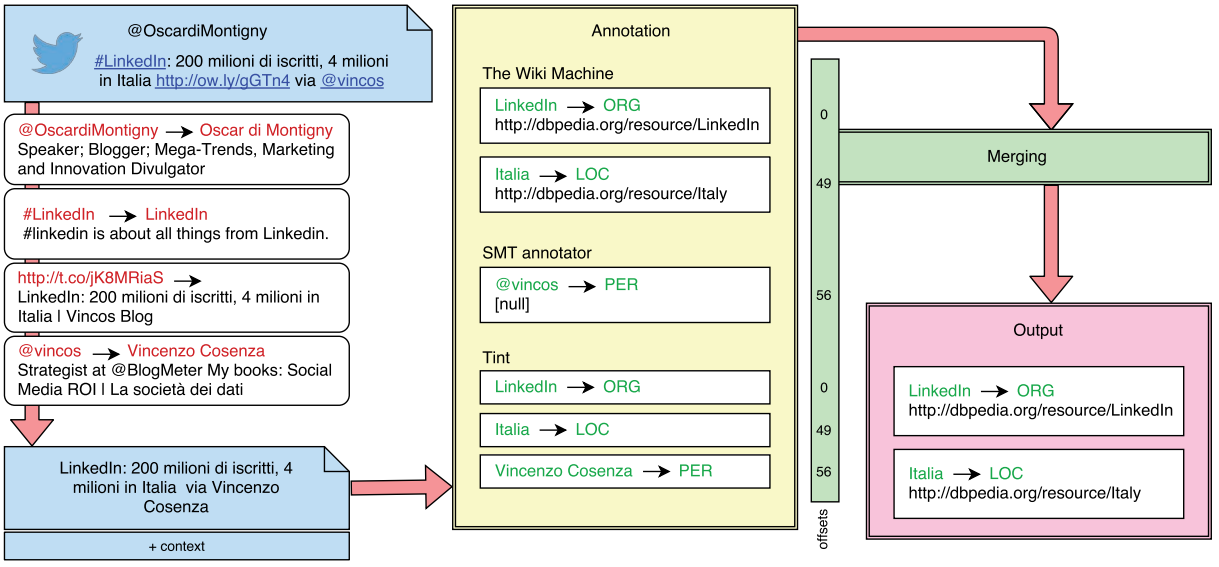


Figure 2: An example of annotation.

the *rule-based SMT annotator* always annotates if the SMT type is person or organization, whereas the *supervised SMT annotator* decides using an SVM classifier trained on the development set of NEEL-IT.

The middle box in Figure 2 shows the entities extracted by each tool: The Wiki Machine recognizes "LinkedIn" as organization and "Italia" as location; SMT identifies "@vincos" as a person; and Tint classifies "LinkedIn" as organization and "Italia" and "Vincenzo Cosenza" as persons.

3.3 Merging

The last part of the pipeline consists in deciding which annotations have to be kept and which ones should be discarded. In addition, the system has to choose how to deal with conflicts (for example inconsistency between the class produced by Tint and the one extracted by The Wiki Machine).

Specifically, the task consists in building a *merger* that chooses at most one NER class (and possibly a compatible DBpedia link) for each offset of the text for which at least one annotator recognized an entity. For instance, in the example of Figure 2, the merger should ignore the annotation of @vincos, as it is non considered a named entity.

As baseline, we first developed a *rule-based merger* that does not discard any annotation and solves conflicts by majority vote or, in the event of a tie, by giving different priorities to the annotations produced by each annotator.[10]

[10]Tint first, followed by The Wiki Machine and SMT.

We then trained a *supervised merger* consisting of a multi-class SVM whose output is either one of the NER categories or a special NONE category, for which case we discard all the annotations for the offset. The classifier is trained on the development tweets provided by the task organizers, using libSVM (Chang and Lin, 2011) with a polynomial kernel and controlling precision/recall via the penalty parameter C for the NONE class. Given an offset and the associated entity annotations we use the following features:

- whether the entity is linked to DBpedia;
- whether the tool x annotated this entity;
- whether the tool x annotated the entity with category y (x can be Tint, SMT, or The Wiki Machine; y can be one of the possible categories, such as person, location, and so on);
- the case of the annotated text (uppercase initials, all uppercase, all lowercase, etc.);
- whether the annotation is contained in a Twitter username and/or in a hashtag;
- whether the annotated text is an Italian common word and/or a known proper name; common words were taken from Morph-It! (see Section 3.1), while proper nouns were extracted from Wikipedia biographies;
- whether the annotated text contains more than one word;
- frequencies of NER categories in the training dataset of tweets.

The result of the merging step is a set of NER and EL annotations as required by the NEEL-IT

task. EL annotations whose DBpedia entities are not part of the English DBpedia were discarded when participating in the task, as for NEEL-IT rules. They were however exploited for placing the involved entities in the same coreference set. The remaining (cross-micropost) coreference annotations for unlinked (NIL) entities were derived with a simple baseline that always put entities in different coreference sets.[11]

4 Results

Table 1 reports on the performances obtained by MicroNeel at the NEEL-IT task of EVALITA 2016, measured using three sets of Precision (P), Recall (R), and F1 metrics (Basile et al., 2016a):

- *mention CEAF* tests coreference resolution;
- *strong typed mention match* tests NER (i.e., spans and categories of annotated entities);
- *strong link match* assesses EL (i.e., spans and DBpedia URIs of annotated entities).

Starting from their F1 scores, an overall F1 score was computed as a weighted sum (0.4 for mention CEAF and 0.3 for each other metric).

MicroNeel was trained on the development set of 1000 annotated tweets distributed as part of the task, and tested on 300 tweets. We submitted three runs (upper part of Table 1) that differ on the techniques used – rule-based vs supervised – for the SMT annotator and the merger:

- *base* uses the rule-based variants of the SMT annotator and the merger;
- *merger* uses the rule-based SMT annotator and the supervised merger;
- *all* uses the supervised variants of the SMT annotator and the merger.

In addition to the official NEEL-IT scores, the lower part of Table 1 reports the result of an ablation test that starts from the *base* configuration and investigates the contributions of different components of MicroNeel: The Wiki Machine (EL), Tint (NER), SMT, the tweet rewriting, and the addition of textual context during preprocessing.

5 Discussion

Contrarily to our expectations, the *base* run using the simpler *rule-based SMT* and *rule-based merger* performed better than the other runs employing supervised techniques. Table 1 shows that the contribution of the *supervised SMT* annotator was null on the test set. The *supervised merger*, on the other hand, is only capable of changing the precision/recall balance (which was already good for the *base* run) by keeping only the best annotations. We tuned it for maximum F1 via cross-validation on the development set of NEEL-IT, but the outcome on the test set was a decrease of recall not compensated by a sufficient increase of precision, leading to an overall decrease of F1.

The ablation test in the lower part of Table 1 shows that the largest drop in performances results from removing The Wiki Machine, which is thus the annotator most contributing to overall performances, whereas SMT is the annotator giving the smallest contribution (which still amounts to a valuable +0.0193 F1). The rewriting of tweet texts accounts for +0.0531 F1, whereas the addition of textual context had essentially no impact on the test set, contrarily to our expectations.

An error analysis on the produced annotations showed that many EL annotations were not produced due to wrong word capitalization (e.g., lower case words not recognized as named entities), although the true-casing performed as part of preprocessing mitigated the problem. An alternative and possibly more robust solution may be to retrain the EL tool not considering letter case.

6 The tool

The MicroNeel extraction pipeline is available as open source (GPL) from the project website.[12] It is written in Java and additional components for preprocessing, annotation, and merging can be easily implemented by implementing an `Annotator` interface. The configuration, including the list of components to be used and their parameters, can be set through a specific JSON configuration file. Extensive documentation will be available soon on the project wiki.

7 Conclusion and Future Work

In this paper we presented MicroNeel, a system for Named Entity Recognition and Entity Linking on Italian microposts. Our approach consists of three main steps, described in Section 3: preprocessing, annotation, and merging. By getting the second best result in the NEEL-IT task at EVALITA

[11] It turned out after the evaluation that the alternative baseline that corefers entities with the same (normalized) surface form performed better on NEEL-IT test data.

[12] `https://github.com/fbk/microneel`

Table 1: MicroNeel performances on NEEL-IT test set for different configurations.

Configuration	Mention CEAF			Strong typed mention match			Strong link match			Overall
	P	R	F1	P	R	F1	P	R	F1	F1
base run	0.514	0.547	0.530	0.457	0.487	0.472	0.567	0.412	0.477	0.4967
merger run	0.576	0.455	0.509	0.523	0.415	0.463	0.664	0.332	0.442	0.4751
all run	0.574	0.453	0.506	0.521	0.412	0.460	0.670	0.332	0.444	0.4736
base - NER	0.587	0.341	0.431	0.524	0.305	0.386	0.531	0.420	0.469	0.4289
base - SMT	0.504	0.525	0.514	0.448	0.468	0.458	0.564	0.372	0.448	0.4774
base - EL	0.487	0.430	0.457	0.494	0.437	0.464	0.579	0.049	0.090	0.3490
base - rewriting	0.554	0.399	0.464	0.492	0.356	0.413	0.606	0.354	0.447	0.4436
base - context	0.513	0.547	0.530	0.453	0.485	0.468	0.566	0.416	0.480	0.4964

2016, we demonstrated that our approach is effective even if it builds on standard components.

Although the task consists in annotating tweets in Italian, MicroNeel is largely agnostic with respect to the language, the only dependencies being the dictionaries used for preprocessing, as both The Wiki Machine and Tint NER support different languages while SMT is language-independent. Therefore, MicroNeel can be easily adapted to other languages without big effort.

MicroNeel is a combination of existing tools, some of which already perform at state-of-the-art level when applied on tweets (for instance, our system got the best performance in the linking task thanks to The Wiki Machine). In the future, we plan to adapt MicroNeel to English and other languages, and to integrate some other modules both in the preprocessing and annotation steps, such the NER system expressly developed for tweets described by Minard et al. (2016).

Acknowledgments

The research leading to this paper was partially supported by the European Union's Horizon 2020 Programme via the SIMPATICO Project (H2020-EURO-6-2015, n. 692819).

References

Pierpaolo Basile, Annalina Caputo, Anna Lisa Gentile, and Giuseppe Rizzo. 2016a. Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian tweets (NEEL-IT) task. In *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli. 2016b. EVALITA 2016: Overview of the 5th evaluation campaign of natural language processing and speech tools for Italian. aAcademia University Press.

Kalina Bontcheva, Leon Derczynski, Adam Funk, Mark A. Greenwood, Diana Maynard, and Niraj Aswani. 2013. TwitIE: An open-source information extraction pipeline for microblog text. In *Recent Advances in Natural Language Processing, RANLP*, pages 83–90. RANLP 2013 Organising Committee / ACL.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195.

Anne-Lyse Minard, Mohammed R.H. Qwaider, and Bernardo Magnini. 2016. FBK-NLP at NEEL-IT: Active Learning for Domain Adaptation. In *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. 2016. Linking knowledge bases to social media profiles. `http://alignments.futuro.media/`.

Alessio Palmero Aprosio and Claudio Giuliano. 2016. The Wiki Machine: an open source software for entity linking and enrichment. *ArXiv e-prints*.

Alessio Palmero Aprosio and Giovanni Moretti. 2016. Italy goes to Stanford: a collection of CoreNLP modules for Italian. *ArXiv e-prints*.

Alessio Palmero Aprosio, Claudio Giuliano, and Alberto Lavelli. 2013. Automatic expansion of DBpedia exploiting Wikipedia cross-language information. In *Proceedings of the 10th Extended Semantic Web Conference*, pages 397–411. Springer.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! a free corpus-based morphological resource for the Italian language. *Corpus Linguistics 2005*, 1(1).

sisinflab: an ensemble of supervised and unsupervised strategies for the NEEL-IT challenge at Evalita 2016

Vittoria Cozza, Wanda La Bruna, Tommaso Di Noia
Polytechnic University of Bari
via Orabona, 4, 70125, Bari, Italy
{vittoria.cozza, wanda.labruna, tommaso.dinoia}@poliba.it

Abstract

English. This work presents the solution adopted by the sisinflab team to solve the task NEEL-IT (Named Entity rEcognition and Linking in Italian Tweets) at the Evalita 2016 challenge. The task consists in the annotation of each named entity mention in a Twitter message written in Italian, among *characters*, *events*, *people*, *locations*, *organizations*, *products* and *things* and the eventual linking when a corresponding entity is found in a knowledge base (e.g. DBpedia). We faced the challenge through an approach that combines unsupervised methods, such as DBpedia Spotlight and word embeddings, and supervised techniques such as a CRF classifier and a Deep learning classifier.

Italiano. *Questo lavoro presenta la soluzione del team sisinflab al task NEEL-IT (Named Entity rEcognition and Linking in Italian Tweets) di Evalita 2016. Il task richiede il riconoscimento e l'annotazione del testo di un messaggio di Twitter in Italiano con entità nominate quali personaggi, eventi, persone, luoghi, organizzazioni, prodotti e cose e eventualmente l'associazione di queste entità con la corrispondente risorsa in una base di conoscenza quale, DBpedia. L'approccio proposto combina metodi non supervisionati quali DBpedia Spotlight e i word embeddings, e tecniche supervisionate basate su due classificatori di tipo CRF e Deep learning.*

1 Introduction

In the interconnected world we live in, the information encoded in Twitter streams represents a valuable source of knowledge to understand events, trends, sentiments as well as user-behaviors. While processing these small text messages a key role is played by the entities which are named within the Tweet. Indeed, whenever we have a clear understanding of the entities involved in a context, a further step can be done by semantically enriching them via side information available, e.g., in the Web. To this aim, pure NER techniques show their limits as they are able to identify the category an entity belongs to but they cannot be used to find further information that can be used to enrich the description of the identified entity and then of the overall Tweet. This is the point where Entity Linking starts to play its role. Dealing with Tweets, as we have very short messages and texts with little context, the challenge of Named Entity Linking is even more tricky as there is a lot of noise and very often text is semantically ambiguous. A number of popular challenges on the matter currently exists, as those included in the SemEval series on the evaluations of computational semantic analysis systems[1] for English, the CLEF initiative[2] that provides a cross-language evaluation forum or Evalita[3] that aims to promote the development of language and speech technologies for the Italian language.

Several state of the art solutions have been proposed for entity extraction and linking to a knowledge base (Shen et al., 2015) and many of them make use of the datasets available as Linked (Open) Data such as DBpedia or Wikidata (Gangemi, 2013). Most of these tools expose the best performances when used with long texts. Anyway, those approaches that perform well on newswire domain do not work as well in a microblog scenario. As analyzed in (Derczynski et al., 2015), conventional tools (i.e., those trained

[1] https://en.wikipedia.org/wiki/SemEval
[2] http://www.clef-initiative.eu/
[3] http://www.evalita.it/

on newswire) perform poorly in this genre, and thus microblog domain adaptation is crucial for good NER. However, when compared to results typically achieved on longer news and blog texts, state-of-the-art tools in microblog NER still reach bad performance. Consequently, there is a significant proportion of missed entity mentions and false positives. In (Derczynski et al., 2015), the authors also show which tools are possible to extend and adapt to Twitter domain, for example DBpedia Spotlight.The advantage of Spotlight is that it allows users to customize the annotation task. In (Derczynski et al., 2015) the authors show Spotlight achieves 31.20% of F1 over a Twitter dataset.

In this paper we present the solution we propose for the NEEL-IT task (Basile et al., 2016b) of Evalita 2016 (Basile et al., 2016a). The task consists of annotating each named entity mention (characters, events, people, locations, organizations, products and things) in an Italian Tweet text, linking it to DBpedia nodes when available or labeling it as NIL entity otherwise. The task consists of three consecutive steps: (1) extraction and typing of entity mentions within a tweet; (2) linking of each textual mention of an entity to an entry in the canonicalized version of DBpedia 2015-10 representing the same "real world" entity, or NIL in case such entry does not exist; (3) clustering of all mentions linked to NIL. In order to evaluate the results the TAC KBP scorer[4] has been adopted. Our team solutions faces the above mentioned challenges by using an ensemble of state of the art approaches.

The remainder of the paper is structured as follows: in Section 2 we introduce our strategy that combines DBpedia Spotlight-based and a machine learning-based solutions, detailed respectively in Section 2.1 and Section 2.2. Section 3 reports and discusses the challenge results.

2 Description of the system

The system proposed for entity boundary and type extraction and linking is an ensemble of two strategies: a DBpedia Spotligth[5]-based solution and a machine learning-based solution, that exploits Stanford CRF[6] and DeepNL[7] classifiers. Before applying both approaches we pre-processed the tweets used in the experiments, by doing: (1) data cleaning consisting of replacing URLs with the keyword `URL` as well emoticons with `EMO`; This has been implemented with ad hoc rules; (2) sentence splitter and tokenizer, implemented by the well known linguistic pipeline available for the Italian language: "openNLP"[8], with its corresponding binary models[9].

2.1 Spotlight-based solution

DBpedia Spotlight is a well known tool for entity linking. It allows a user to automatically annotate mentions of DBpedia resources in unstructured textual documents.

• Spotting: recognizes in a sentence the phrases that may indicate a mention of a DBpedia resource.

• Candidate selection: maps the spotted phrase to resources that are candidate disambiguations for that phrase.

• Disambiguation: uses the context around the spotted phrase to decide for the best choice amongst the candidates.

In our approach we applied DBpedia Spotlight (J. et al., 2013) in order to identify mention boundaries and link them to a DBpedia entity. This process makes possible to identify only those entities having an entry in DBpedia but it does not allow a system to directly identify entity types. According to the challenge guideline we required to identify entities that fall into 7 categories: `Thing`, `Product`, `Person`, `Organization`, `Location`, `Event`, `Character` and their subcategories. In order to perform this extra step, we used the "type detection" module, as shown in Figure 1 which makes use of a `SPARQL` query to extract ontological information from DBpedia. In detail we match the name of returned classes associated to an entity with a list of keywords related to the available taxonomy: Place, Organization (or Organisation), Character, Event, Sport, Disease, Language, Person, Music Group, Software, Service, Film, Television, Album, Newspaper, Electronic Device. There are three possible outcomes: no match, one match, more than one match. In the case we find no match we discard the entity while in case we have more than one match we choose

[4] https://github.com/wikilinks/neleval/wiki/Evaluation

[5] urlhttps://github.com/dbpedia-spotlight/dbpedia-spotlight

[6] http://nlp.stanford.edu/software/CRF-NER.shtml

[7] https://github.com/attardi/deepnl

[8] `https://opennlp.apache.org/index.html`

[9] `https://github.com/aciapetti/opennlp-italian-models/tree/master/models/it`

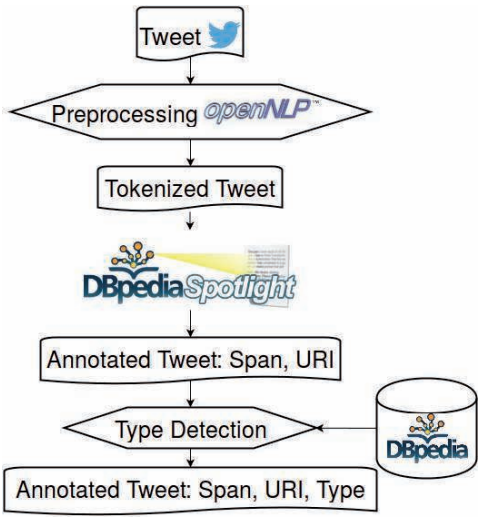


Figure 1: Spotlight based solution

the most specific one, according the NEEL-IT taxonomy provided for the challenge. Once we have an unique match we return the entity along with the new identified type.

Since DBpedia returns entities classified with reference to around 300 categories, we process the annotated resources through the Type Detection Module to discard all those entities not falling in any of the categories of the NEEL-IT taxonomy. Over the test set, after we applied the Ontology-based type detection module, we discarded 16.9% of returned entities. In this way, as shown in Figure 1, we were able to provide an annotation (span, uri, type) as required by the challenge rules.

2.2 Machine learning based solution

As summarized in Figure 2, we propose an ensemble approach that combines unsupervised and supervised techniques by exploiting a large dataset of unannotated tweets, Twita (Basile and Nissim, 2013) and the DBpedia knowledge base. We used a supervised approach for entity name boundary and type identification, that exploits the challenge data. Indeed the challenge organizers provided a training dataset consisted of 1,000 tweets in italian, for a total of 1,450 sentences. The training dataset were annotated with 801 gold annotations. Overall 526 over 801 were entities linked to a unique resource on DBpedia, the other were linked to 255 NIL clusters. We randomly split this training dataset in `new_train` (70%) and `validation` (30%) set. In Table 1 we show the number of mentioned entities classified with reference to their corresponding categories. We then pre-processed the `new_train` and the `validation` sets with the approach

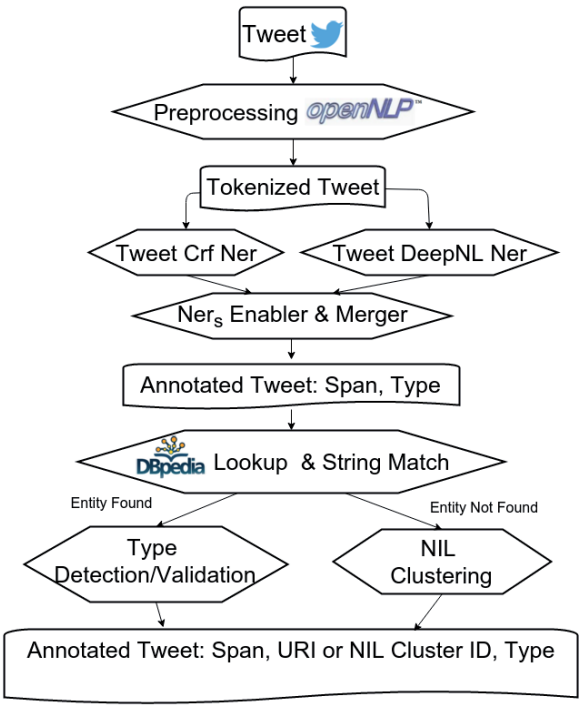


Figure 2: Machine Learning based solution

shortly described in Section 2 thus obtaining a corpus in IOB2-notation. The annotated corpus was then adopted for training and evaluating two classifiers, Stanford CRF(Finkel et al., 2005) and DeepNL(Attardi, 2015) as shown in Figure 2, in order to detect the span and the type of entity mention in the text.

The module *NERs Enabler & Merger* aims to enabling the usage of one or both classifiers. When them both are enabled there can be a mention overlap in the achieved results. In order to avoid overlaps we exploited regular expressions. In particular, we merged two or more mentions when they are consecutive, and we choose the largest span mention when there is a containment. While with Spotlight we are allowed to find linked entities only, with this approach we can detect both entities that matches well known DBpedia resources and those that have not been identified by Spotlight (NIL). In this case given an entity spot, for entity linking we exploited DBpedia Lookup and string matching between mention spot and the labels associated to DBpedia entities. In this way we were able to find both entities along with their URIs, plus several more NIL entities. At this point, for each retrieved entity we have the span, the type (multiple types if CRF and DeepNL disagree) and the URI (see Figure 2) so we use a type detection/validation module for assigning the correct type to an entity. This module uses ad hoc

	#tweets	Character	Event	Location	Organization	Person	Product	Thing
Training set	1,450	16	15	122	197	323	109	20
New_train set	1,018	6	10	82	142	244	68	12
Validation set	432	10	5	40	55	79	41	8

Table 1: Dataset statistics

rules for combining types obtained from the classifier with CRF, DeepNL classifier if they disagree and from DBpedia entity type, when the entity is not NIL. For all NIL entities, finally we cluster them, as required by the challenge, by simply clustering entities with the same type and surface form. We consider also surface forms that differ in case (lower and upper).

CRF NER. The Stanford Named Entity Recognizer is based on the Conditional Random Fields (CRF) statistical model and uses Gibbs sampling for inference on sequence models(Finkel et al., 2005). This tagger normally works well enough using just the form of tokens as feature. This NER is a widely used machine learning-based method to detect named entities, and is distributed with CRF models for English newswire text. We trained the CRF classifier for Italian tweets with the `new_train` data annotated with IOB notation, then we evaluate the results across the validation data, results are reported in Table 2. The results provided follow the CoNLL NER evaluation (Sang and Meulder, 2003) format that evaluates the results in term of Precision (**P**) and Recall (**R**). The F-score (**F1**) corresponds to the `strong_typed_mention_match` in the TAC scorer. A manual error analysis showed that even

Entity	P	R	F1	TP	FP	FN
LOC	0.6154	0.4000	0.4848	16	10	24
ORG	0.5238	0.2000	0.2895	11	10	44
PER	0.4935	0.4810	0.4872	38	39	41
PRO	0.2857	0.0488	0.0833	2	5	39
Totals	0.5115	0.2839	0.3651	67	64	169

Table 2: CRF NER over the validation set

when mentions are correctly detected, types are wrongly identified. This is due of course to language ambiguity in a sentence. As an example, for a NER it is often hard to disambiguate between a person and an organization, or an event and a products are not. For this reason we applied a further type detection and validation module which allowed to combine, by ad hoc rules, the results obtained by the classifiers and the Spotlight-based approach previously described.

DeepNL NER. DeepNL is a Python library for Natural Language Processing tasks based on a Deep Learning neural network architecture. The library currently provides tools for performing part-of-speech tagging, Named Entity tagging and Semantic Role Labeling. External knowledge and Named Entity Recognition World knowledge is often incorporated into NER systems using gazetteers: categorized lists of names or common words. The Deep Learning NLP NER exploits suffix and entities dictionaries and it uses word embedding vectors as main feature. The entity dictionary has been created by using the entity mention from the training set, and also the locations mentions provided by SENNA[10]. The suffix dictionary has been extracted as well from the training set with ad hoc scripts. Word embeddings were created using the Bag-of-Words (CBOW) model by (Mikolov et al., 2013) of dimension 300 with a window size of 5. In details we used the software word2vec available from `https://code.google.com/archive/p/word2vec/`, over a corpus of above 10 million of unlabeled tweets in Italian. In fact, the corpus consists of a collection of the Italian tweets produced in April 2015 extracted from the Twita corpus (Basile and Nissim, 2013) plus the tweets both from `dev` and `test` sets provided by the NEEL-IT challenge, all them pre-processed through our data preprocessing module, with a total of 11.403.536 sentences. As shown in Figure 3, we trained a DeepNL classifier for Italian tweets with the `new_train` data annotated with IOB-2 notation then we evaluate the results across the validation data. Over the validation set we obtained an accuracy of 94.50%. Results are reported in Table 3.

Entity	P	R	F1	Correct
EVE	0	0	0	1
LOC	0.5385	0.1750	0.2642	13
ORG	0.4074	0.2	0.2683	27
PER	0.6458	0.3924	0.4882	48
PRO	0.4375	0.1707	0.2456	16
Totals	0.5333	0.2353	0.3265	104

Table 3: DeepNL NER over the validation set

2.3 Linking

For the purpose of accomplish the linking sub task, we investigated if a given spot, identified by the machine learning approach as an entity, has a cor-

[10] `http://ronan.collobert.com/senna/`

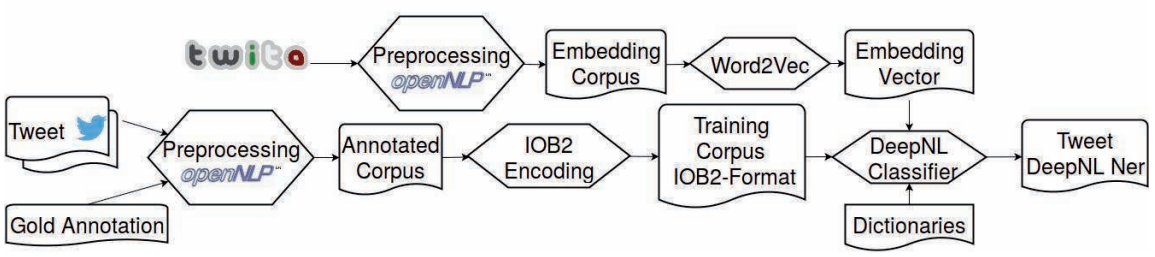


Figure 3: DeepNL: Training phase

responding link in DBpedia. A valid approach to link the names in our datasets to entities in DBpedia is represented by DBpedia Lookup[11] (Bizer et al., 2009) which behaves as follows:

candidate entity generation. A dictionary is created via a Lucene index. It is built starting from the values of the property `rdfs:label` associated to a resource. Very interestingly, the dictionary takes into account also the `Wikipedia:Redirect`[12] links.

candidate entity ranking. Results computed via a lookup in the dictionary are then weighted combining various string similarity metrics and a PageRank-like relevance rankings.

unlinkable mention prediction. The features offered by DBpedia Lookup to filter out resources from the candidate entities are: (i) selection of entities which are instances of a specific class via the `QueryClass` parameter; (ii) selection of the top N entities via the `MaxHits` parameter.

As for the last step we used the Type Detection module introduced above, to select entities belonging only to those classes representative of the interest domain. We implemented other filters to reduce the number of false positives in the final mapping. As an example, we discard the results for the case of Person entity, unless the mention exactly matches the entity name. As a plus, for linking, we also used a dictionary made from the training set, where for a given surface form and a type it returns a correspondent URI, if already available in the labeled data.

Computing canonicalized version. The link results obtained through Spotlight and Lookup or string match, refer to the Italian version of DBpedia. In order to canonicalized version as required by the task, we automatically found the corresponding canonicalized resource link for each Italian resource by means of the `owl:sameAs` property.

As an example the triple `dbpedia:Multiple_endocrine_neoplasia> owl:sameAs <http://it.dbpedia.org/resource/Neoplasia_endocrina_multipla>` maps the Italian version of *Neoplasia_endocrina_multipla* to its canonicalized version. In a few cases we were not able to perform the match.

3 Results and Discussion

In this section we report the results over the gold test set distibuted to the challenge participants, considering first 300 tweets only.

In order to evaluate the task results, the 2016 NEEL-it challenge uses the TAC KBP scorer[13]. TAC KBP scorer evaluates the results according to the following metrics: **mention_ceaf**, **strong_typed_mention_match** and **strong_linked_match**.

The overall score is a weighted average score computed as:

$$\textbf{score} = 0.4 \cdot \textbf{mention\_ceaf} + 0.3 \cdot \textbf{strong\_link\_match} + \\ +0.3 \cdot \textbf{strong\_typed\_mention\_match}$$

Our solution combines approaches presented in Section 2.1 and Section 2.2. For the 3 runs submitted for the challenge, we used the following configurations: **run1** Spotlight with results coming from both CRF and DeepNL classifiers; **run2** without CRF; **run3** without DeepNL.

As for CRF and DeepNL classifiers, we used a model trained with the whole training set provided by the challenge organizers. In order to ensemble the systems output we applied again the NERs Enabler & Merger module, presented in Section 2.2 that aims to return the largest number of entity annotations identified by the different systems without overlap. If one mention has been identified with more then one approach, and they disagree about the type, that returned by the Spotlight approach is chosen. Results for the different runs are shown in Table 4 together with the results of

[11] `https://github.com/dbpedia/lookup`

[12] `https://en.wikipedia.org/wiki/Wikipedia:Redirect`

[13] https://github.com/wikilinks/neleval/wiki/Evaluatio

System	mention_ceaf	strong_typed_ _mention_match	strong_link_match	final_score
Spotlight-based	0.317	0.276	0.340	0,3121
run1	0.358	0.282	0.38	0.3418
run2	0.34	0.28	0.381	0.3343
run3	0.358	0.286	0.376	0.3418
Best Team	0.561	0.474	0.456	0.5034

Table 4: Challenge results

the best performing team of the challenge. In order to evaluate the contribution of the Spotlight-based approach to the final result, we evaluated the **strong_link_match** considering only the portion of link-annotation due to this approach over the challenge test set, see Table 5. We had a total of 140 links to Italian DBpedia, then following the approach described in Section 2.3 we obtained 120 links, 88 of which were unique. It was not possible to convert into DBpedia canonicalized version 20 links. Final results are summarized in Table 5. Looking at the Spotlight-based solution (row 1),

System	P	R	F1
Spotlight-based	0.446	0.274	0.340
run1	0.577	0.28	0.380

Table 5: **strong_link_match** over the challenge gold test set (300 tweets)

compared with the ensemble solution (row 2) results, we saw a performance improvement. This means that machine learning-based approach allowed to identify and link entities that were not detected by Spotlight thus improving precision results. Moreover, combining the two approaches allowed the system, at the step of merging the overlapping span, for a better identification of entities. This behavior lead sometime to delete correct entities, but also to correctly detect errors produced by the Spotlight-based approach and, more generally, it improved recall results.

In the current entity linking literature, mention detection and entity disambiguation are frequently cast as equally important but distinct problems. However, in this task, we find that mention detection often represents a bottleneck. In **mention_ceaf** detection, our submission results show that CRF NER worked slightly better then Deep NER, as already showed in the experiments over the validation set in Section 2.2. Anyway according to experiments in (Derczynski et al., 2015) with a similar dataset and a smaller set of entities, we expected better results from CRF NER. A possible explanation is that errors are due also to the larger number of types to detect as well as to a wrong recombination of overlapping mentions, that has been addressed using simple heuristics.

References

G. Attardi. 2015. Deepnl: a deep learning nlp pipeline. *Workshop on Vector Space Modeling for NLP, NAACL*.

P. Basile and M. Nissim. 2013. Sentiment analysis on italian tweets. In *Proc. of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*.

P. Basile, A. Caputo, A. L. Gentile, and G. Rizzo. 2016a. Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) Task. In *Proc. of the 5th EVALITA*.

P. Basile, F. Cutugno, M. Nissim, V. Patti, and R. Sprugnoli. 2016b. EVALITA 2016: Overview of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Academia University Press.

C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. 2009. {DBpedia} - a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154 – 165.

L. Derczynski, D. Maynard, G. Rizzo, M. van Erp, G. Gorrel, R. Troncy, J. Petrak, and K. Bontcheva. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49.

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of the 43rd ACL '05*.

A. Gangemi. 2013. A comparison of knowledge extraction tools for the semantic web. In *Proc. of ESWC*.

J., M. Jakob, C. Hokamp, and P. N. Mendes. 2013. Improving efficiency and accuracy in multilingual entity extraction. In *Proc. of the 9th I-Semantics*.

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *In Advances in Neural Information Processing Systems*, pages 3111–3119.

E. F. Tjong Kim Sang and F. De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proc. of 7th CONLL*, pages 142–147.

W. Shen, J. Wang, and J. Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on KDE*, 27(2):443–460.

FBK-NLP at NEEL-IT: Active Learning for Domain Adaptation

Anne-Lyse Minard[1,2], Mohammed R. H. Qwaider[1], Bernardo Magnini[1]
[1] Fondazione Bruno Kessler, Trento, Italy
[2] Dept. of Information Engineering, University of Brescia, Italy
{minard,qwaider,magnini}@fbk.eu

Abstract

English. In this paper we present the FBK-NLP system which participated to the NEEL-IT task at Evalita 2016. We concentrated our work on domain adaptation of an existed Named Entity Recognition tool. Particularly, we created a new annotated corpus for the NEEL-IT task using an Active Learning method. Our system obtained the best results for the task of Named Entity Recognition, with an F1 of 0.516.

Italiano. *In questo articolo descriviamo il sistema FBK-NLP con il quale abbiamo partecipato al task NEEL-IT a Evalita 2016. Ci siamo concentrati sull'adattamento di un sistema per il riconoscimento di entità al dominio dei tweets. In particolare, abbiamo creato un nuovo corpus usando una metodologia basata su Active Learning. Il sistema ha ottenuto i risultati migliori sul sottotask di riconoscimento delle entità, con una F1 di 0,516.*

1 Introduction

This paper describes the FBK-NLP system which participated to the NEEL-IT task at EVALITA 2016 (Basile et al., 2016). The NEEL-IT task focuses on Named Entity Linking in tweets in Italian. It consists in three steps: Named Entity Recognition and Classification (NER) in 7 classes (person, location, organization, product, event, thing and character); the linking of each entity to an entry of DBpedia; the clustering of the entities. Our participation to the task was mainly motivated by our interest in experimenting on the application of Active Learning (AL) for domain adaptation, in particular to adapt a general purpose Named Entity Recognition system to a specific domain (tweets) by creating new annotated data.

The system follows 3 steps: entity recognition and classification, entity linking to DBpedia and clustering. Entity recognition and classification is performed by the EntityPro module (Pianta and Zanoli, 2007), which is based on machine learning and uses the SVM algorithm. Entity linking is performed using the named entity disambiguation module developed within the NewsReader project for several languages including Italian. In addition we used the Alignments dataset (Nechaev et al., 2016), a resource which provides links between Twitter profiles and DBpedia. Clustering step is string-based, i.e. two entities are part of the same cluster if they are equal.

The paper is organized as follows. In Section 2 we present the domain adaptation of the Named Entity Recognition tool using Active Learning. Then in Section 3 we describe the system with which we participated to the task and in Section 4 the results we obtained as well as some further experiments. Finally we conclude the paper with a discussion in Section 5.

2 Domain Adaptation for NER

We have at our disposal a system for Named Entity Recognition and Classification, a module of the TextPro pipeline (Pianta et al., 2008) called EntityPro (Pianta and Zanoli, 2007), which works for 4 named entity categories in the news domain. It is trained on the publicly available Italian corpus I-CAB (Magnini et al., 2006). I-CAB is composed of news articles from the regional newspaper "L'Adige", is annotated with person, organization, location and geo-political entities, and was used for the Named Entity Recognition task at Evalita 2007 and 2009.[1] However, no annotated data are available for the task of NER in tweets for

[1] www.evalita.it/

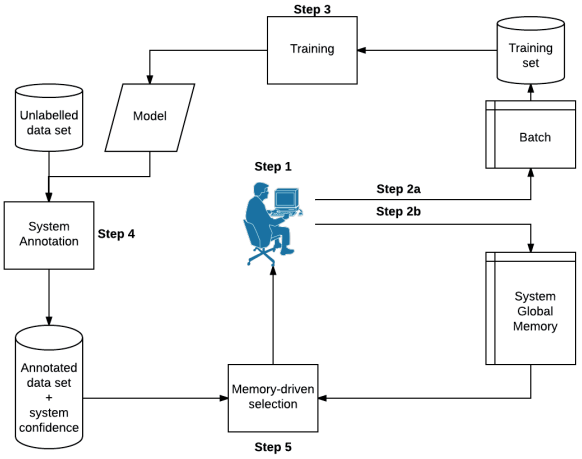


Figure 1: Architecture of the TextPro-AL platform

Italian.

As we were interested in applying Active Learning (AL) methods to the production of training data, we decided to annotate manually our own set of domain specific training data using AL method.[2] Active Learning is used in order to select the most informative examples to be annotated, instead of selecting random examples.

We exploited TextPro-AL (Magnini et al., 2016), a platform which integrates a NLP pipeline, i.e. TextPro (Pianta et al., 2008), with a system of Active Learning and an annotation interface based on MTEqual (Girardi et al., 2014). TextPro-AL enables for a more efficient use of the time of the annotators.

2.1 The TextPro-AL platform

The architecture of the TextPro-AL platform is represented in Figure 1. The AL cycle starts with an annotator providing supervision on a tweet automatically tagged by the system (step 1): the annotator is asked to revise the annotation in case the system made a wrong classification. At step 2a the annotated tweet is stored in a batch, where it is accumulated with other tweets for re-training, and, as a result, a new model (step 3) is produced. This model is then used to automatically annotate a set of unlabeled tweets (step 4) and to assign a confidence score[3] to each annotated tweet. At step 2b the manually annotated tweet is stored in the Global Memory of the system with the information about the manual revision. At step 5 a single tweet is selected from the unlabeled dataset through a specific selection strategy (see Algorithm 1). The selected tweet is removed from the unlabeled set and is given for revision to the annotator.

The Global Memory contains the revision done by the annotator for each tweet. In particular we are interested in the entities wrongly annotated by the system, which are used to select new tweets to be annotated. Each entity (or error) saved in the memory is used up to 6 times in order to select new tweets. From the unlabeled dataset, the system selects the most informative instance (i.e. with the lowest confidence score) that contains one of the errors saved in the Global Memory (GM). The selection strategy is detailed in Algorithm 1. In a first step the system annotates the tweets of the unlabeled dataset. Then the tweets are sorted from the most informative to the less informative and browsed. The first tweet in the list that contains an error saved in the GM is selected to be revised by the annotator. If no tweets are selected through this process, the system picks one tweet randomly.

Algorithm 1: Algorithm of the selection strategy

```
Data: NESet = {NE_1 ... NE_n}
begin
    NESortedList ←
    getMostInformativeInstances(NESet);
    repeat
        instance, sample ←
        NESortedList.next();
        if inMemory(instance) and
        revised(instance) then
            return sample;
    until NESortedList.hasNext();
    return getRandomSample(NESet);
```

2.2 Available Data

As unlabeled database of tweets in the AL process we used around 8,000 tweets taken from the development set of Sentipolc 2014[4] (Basile et al., 2014) and the Twita corpus[5] (Basile and Nissim, 2013).

[2]The annotated data made available by the organizers of the task were used partly as test data and partly as a reference for the annotators (see Section 2.2).

[3]The confidence score is computed as the average of the margin estimated by the SVM classifier for each entity.

[4]http://www.di.unito.it/~tutreeb/sentipolc-evalita14/tweet.html

[5]http://valeriobasile.github.io/twita/about.html

class	AL tweets	NEEL-IT dev			news corpus
		test 70%	dev 30%	total	
# sent/tweets	2,654	700	300	1,000	458
# tokens	49,819	13,283	5,707	18,990	8,304
Person	1628	225	90	315	293
Location	343	89	43	132	115
Organization	723	185	63	248	224
Product	478	67	41	108	-
Event	133	12	3	15	-
Thing	15	15	4	19	-
Character	50	15	1	16	-

Table 1: Statistics about the used datasets. The numbers of tokens for the tweets are computed after the tokenizaion, i.e. the hashtags and aliases can be split in more than one token and the emoji are composed by several tokens (see Section 3.1).

The development data provided by the NEEL-IT organizers is composed by 1000 annotated tweets. We split it in two parts: 30% for development (used mainly as a reference for the annotators) and 70% for evaluation (referred to as *test 70%*).

We decided to retrain EntityPro using a smaller training set to be able to change the behavior of the model more quickly. In particular we used a sub-part of the training data used by EntityPro, i.e. 6.25% of the training set of the NER task at Evalita 2007,[6] for a total of 8,304 tokens (referred to as *news corpus* in the remainder of the paper).

In order to determine the portion to be used, we tested the performance of EntityPro using as training data different portions of the corpus (50%, 25%, 12.5% and 6.25%) on *test 70%*. The best results were obtained using 6.25% of the corpus (statistics about this corpus is given in Table 1).

2.3 Manual Annotation of Training Data with TextPro-AL

In our experimentation with TextPro-AL for domain adaptation we built the first model using the *news corpus* only. Evaluated on *test 70%*, it reached an F1 of 41.62 with a precision of 54.91 and a recall of 33.51. It has to be noted that with this model only 3 categories of entities can be recognized: person, location and organization. Then every time that 50 new tweets were annotated, the system was retrained and evaluated on the *test 70%* corpus. The learning curves of the system are presented in Figure 2. In total we were able to manually annotate 2,654 tweets for a total of 3,370 entities (we will refer to this corpus as *AL tweets*), which allowed us to obtain an F1 of 53.22 on *test 70%*. Statistics about the corpus are presented in Table 1.

3 Description of the system

3.1 Entity Recognition and Classification

The preprocessing of the tweets is done using the TextPro tool suite[7] (Pianta et al., 2008), in particular using the tokenizer, the PoS tagger and the lemmatizer. The rules used by the tokenizer have been lightly adapted for the processing of tweets, for example to be able to split Twitter profile names and hashtags in small units. The PoS tagger and the lemmatizer have been used as they are, without any adaptation.

In order to avoid some encoding problems we replaced all the emoji by their Emoji codes (e.g. :confused_face:) using the python package emoji 0.3.9.[8]

The task of entity recognition and classification is performed using an adapted version of the EntityPro module (Pianta and Zanoli, 2007). EntityPro performs named entity recognition based on machine learning, using an SVM algorithm and the Yamcha tool (Kudo and Matsumoto, 2003). It exploits a rich set of linguistic features, as well as gazetteers. We added to the features an orthographic feature (capitalized word, digits, etc.) and bigrams (the first two characters and the last two).

The classifier is used in a one-vs-rest multi-classification strategy. The format used for the

[6] http://www.evalita.it/2007/tasks/ner

[7] http://textpro.fbk.eu/

[8] http://pypi.python.org/pypi/emoji/

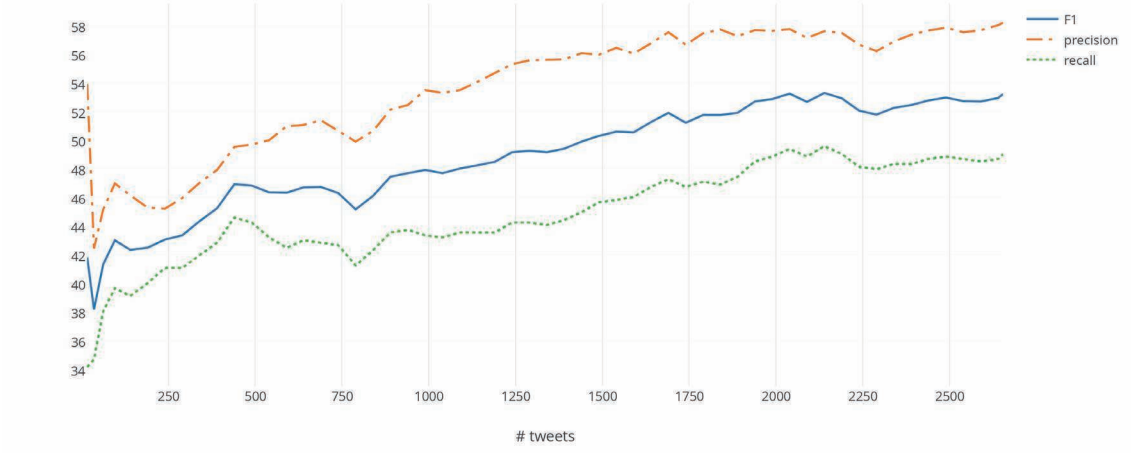


Figure 2: Learning curves of the system (recall, precision and F1)

annotation is the classic IOB2 format. Each token is labeled either as B- followed by the entity class (person, location, organization, product, event, thing or character) for the first token of an entity, I- followed by the entity class for the tokens inside an entity or O if the token is not part of an entity.

3.2 Entity Linking

Entity Linking is performed using the Named Entity Disambiguation (NED) module developed within the NewsReader Project[9] supplemented with the use of a resource for Twitter profiles linking. The NED module is a wrapper around DBpedia spotlight developed within NewsReader and part of the ixa-pipeline.[10] Each entity recognized by the NER module is sent to DBpedia Spotlight which returns the most probable URI if the entity exists in DBpedia.

The tweets often contain aliases, i.e. user profile names, which enable the author of the tweet to refer to other Twitter users. For example @edoardofasoli and @senatoremonti in the following tweet: *@edoardofasoli @senatoremonti Tutti e due.* In order to identify the DBpedia links of the aliases in the tweets we used the Alignments dataset (Nechaev et al., 2016). The Alignments dataset is built from the 2015-10 edition of English DBpedia, which contains DBpedia links aligned with Twitter profiles. It has 920,625 mapped DBpedia entries to their corresponding user profile(s) with a confidence score.

A procedure is built to query Twitter to get the Twitter profile id from the alias of a user, then query the Alignments dataset to get the corresponding DBpedia link if it exists.

3.3 Clustering

The clustering task aims at gathering the entities referring to the same instance and at assigning to them an identifier, either a DBpedia link or a corpus based identifier. We performed this task applying a basic string matched method, i.e. we consider that two entities are part of the same cluster if their strings are the same.

4 Results

We submitted 3 runs to the NEEL-IT task; they differ from the data included in the training dataset of EntityPro:

- Run 1: *news corpus* and *AL tweets*
- Run 2: *news corpus*, *AL tweets* and NEEL-IT devset
- Run 3: *AL tweets* and NEEL-IT devset

The official results are presented in the first part of Table 2. Our best performance is obtained with the run 3, with a final score of 0.49.

[9] http://www.newsreader-project.eu/

[10] https://github.com/ixa-ehu/ixa-pipe-ned

runs	training set	tagging	linking	clustering	final score
run 1	*news corpus* + *AL tweets*	0.509	0.333	0.574	0.4822
run 2	*news corpus* + *AL tweets* + NEEL-IT devset	0.508	0.346	0.583	0.4894
run 3	*AL tweets* + NEEL-IT devset	0.516	0.348	0.585	**0.4932**
run 4*	*AL tweets* + NEEL-IT devset	0.517	0.355	0.590	**0.4976**
run 5*	*news corpus*	0.378	0.298	0.473	0.3920
run 6*	NEEL-IT devset	0.438	0.318	0.515	0.4328
run 7*	*news corpus* + NEEL-IT devset	0.459	0.334	0.541	0.4543

Table 2: Results of the submitted runs (runs 1 to 3) and of some further experiments (runs 4 to 8). The official task metrics are "strong_typed_mention_match", "strong_link_match" and "mention_ceaf", and refer to "tagging", "linking" and "clustering" respectively.

After the evaluation period, we have run further experiments, which are marked with an asterisk in Table 2. The run 4 is a version of run 3 in which we have removed the wrong links to the Italian DBpedia (URIs of type `http://it.dbpedia.org/`). For runs 5, 6 and 7, EntityPro is trained using the *news corpus* alone, the NEEL-IT devset, and both respectively.

In Table 3, we present the performances of our systems in terms of precision, recall and F1 for the subtask of named entity recognition and classification. We observed that using the NEEL-IT devset the precision of our system increased, instead using the news corpus the recall increased.

	precision	recall	F1
run 1	0.571	0.459	0.509
run 2	0.581	0.451	0.508
run 3	0.598	0.454	0.516

Table 3: Results for the task of named entity recognition and classification

5 Discussion

We have described our participation to the NEEL-IT task at Evalita 2016. Our work focused on the task of named entity recognition, for which we get the best results. We were interested in the topic of domain adaptation. The domain adaptation includes two aspects: the type of the documents and the named entity classes of interest. Using EntityPro, an existing NER tool, and the TextPro-AL platform, we created a training dataset for NER in tweets, for the 7 classes identified in the task.[11] With this new resource our system obtained an F1 of 0.516 for named entity recognition.

Our work has been concentrated on the use of Active Learning for the domain adaptation of a NER system. On the other hand, the Micro-NEEL team (Corcoglioniti et al., 2016) focuses on the task of Entity Linking, using The Wiki Machine (Palmero Aprosio and Giuliano, 2016). We have combined our NER system with the Micro-NEEL system. For the tagging subtask we used the same configuration than run 4 (*AL tweets* + NEEL-IT devset). The results obtained with combination of the two systems are 0.517 for tagging, 0.465 for linking and 0.586 for clustering. The final score is 0.5290, surpassing all the runs submitted to the task.

One of the main difficulty in identifying named entities in tweets is the problem of the splitting of hashtags and aliases (e.g. the identification of *Monti* in *@senatoremonti*). We adapted the TextPro tokenizer to split in small units those sequences of characters, but it works only if the different words are capitalized or separated by some punctuation signs (e.g. _ or -). A more complex approach should be used, using a dictionary to improve the splitting.

Named entity categories covered in this task are seven: person, location, organization, product, event, thing and character. The first three categories are the classical ones and cover the highest number of named entities in several corpora. Table 1 gives us an evidence of the prominence of these three classes. With the AL method we used, we were able to annotate new tweets containing entities of the less represented classes, in particular for product, event and character. However the class thing is still not well represented in our corpus and the classes unbalanced. In the future we plan to add in the TextPro-AL platform the pos-

[11] We will soon make available the new training set from the website of the HLT-NLP group at FBK (`http://hlt-nlp.fbk.eu/`).

sibility for the annotators to monitor the Global Memory used in the AL process in order to give precedence to examples containing entities of not well represented classes.

Acknowledgments

This work has been partially supported by the EUCLIP (EUregio Cross LInguistic Project) project, under a collaboration between FBK and Euregio.[12]

References

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta.

Valerio Basile, Andrea Bolioli, Malvina Nissim, Viviana Patti, and Paolo Rosso. 2014. Overview of the Evalita 2014 SENTIment POLarity Classification Task. In *Proceedings of the 4th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'14)*, Pisa, Italy.

Pierpaolo Basile, Annalina Caputo, Anna Lisa Gentile, and Giuseppe Rizzo. 2016. Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Francesco Corcoglioniti, Alessio Palmero Aprosio, Yaroslav Nechaev, and Claudio Giuliano. 2016. MicroNeel: Combining NLP Tools to Perform Named Entity Detection and Linking on Microposts. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Christian Girardi, Luisa Bentivogli, Mohammad Amin Farajian, and Marcello Federico. 2014. Mt-equal: a toolkit for human assessment of machine translation output. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference System Demonstrations, August 23-29, 2014, Dublin, Ireland*, pages 120–123.

Taku Kudo and Yuji Matsumoto. 2003. Fast Methods for Kernel-based Text Analysis. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 24–31, Stroudsburg, PA, USA.

[12] http://www.euregio.it

Bernardo Magnini, Emanuele Pianta, Christian Girardi, Matteo Negri, Lorenza Romano, Manuela Speranza, Valentina Bartalesi Lenzi, and Rachele Sprugnoli. 2006. I-CAB: the Italian Content Annotation Bank. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC-2006)*.

Bernardo Magnini, Anne-Lyse Minard, Mohammed R. H. Qwaider, and Manuela Speranza. 2016. TEXTPRO-AL: An Active Learning Platform for Flexible and Efficient Production of Training Data for NLP Tasks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*.

Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. 2016. Linking knowledge bases to social media profiles.

Alessio Palmero Aprosio and Claudio Giuliano. 2016. The Wiki Machine: an open source software for entity linking and enrichment. *ArXiv e-prints*.

Emanuele Pianta and Roberto Zanoli. 2007. Entitypro: Exploiting svm for italian named entity recognition. *Intelligenza Artificiale numero speciale su Strumenti per lelaborazione del linguaggio naturale per litaliano EVALITA 2007*, 4(2):69–70.

Emanuele Pianta, Christian Girardi, and Roberto Zanoli. 2008. The TextPro Tool Suite. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco.

Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task

Cristina Bosco
Dip. di Informatica, Università di Torino
bosco@di.unito.it

Fabio Tamburini,
FICLIT, University of Bologna, Italy
fabio.tamburini@unibo.it

Andrea Bolioli
CELI
abolioli@celi.it

Alessandro Mazzei
Dip. di Informatica, Università di Torino
mazzei@di.unito.it

Abstract

English. The increasing interest for the extraction of various forms of knowledge from micro-blogs and social media makes crucial the development of resources and tools that can be used for automatically deal with them. PoSTWITA contributes to the advancement of the state-of-the-art for Italian language by: (a) enriching the community with a previously not existing collection of data extracted from Twitter and annotated with grammatical categories, to be used as a benchmark for system evaluation; (b) supporting the adaptation of Part of Speech tagging systems to this particular text domain.

Italiano. *La crescente rilevanza dell'estrazione di varie forme di conoscenza da testi derivanti da micro-blog e social media rende cruciale lo sviluppo di strumenti e risorse per il trattamento automatico. PoSTWITA si propone di contribuire all'avanzamento dello stato dell'arte per la lingua italiana in due modi: (a) fornendo alla comunità una collezione di dati estratti da Twitter ed annotati con le categorie grammaticali, risorsa precedentemente non esistente, da utlizzare come banco di prova nella valutazione di sistemi; (b) promuovendo l'adattamento a questo particolare dominio testuale dei sistemi di Part of Speech tagging che partecipano al task.*

Authors order has been decided by coin toss.

1 Introduction and motivation

In the past the effort on Part-of-Speech (PoS) tagging has mainly focused on texts featured by standard forms and syntax. However, in the last few years the interest in automatic evaluation of social media texts, in particular from microblogging such as Twitter, has grown considerably: the so-called user-generated contents have already been shown to be useful for a variety of applications for identifying trends and upcoming events in various fields.

As social media texts are clearly different from standardized texts, both regarding the nature of lexical items and their distributional properties (short messages, emoticons and mentions, threaded messages, etc.), Natural Language Processing methods need to be adapted for deal with them obtaining reliable results in processing. The basis for such an adaption are tagged social media text corpora (Neunerdt *et al.*, 2013) for training and testing automatic procedures. Even if various attempts to produce such kind of specialised resources and tools are described in literature for other languages (e.g. (Gimpel *et al.*, 2011; Derczynski *et al.*, 2013; Neunerdt *et al.*, 2013; Owoputi *et al.*, 2013)), Italian currently completely lacks of them both.

For all the above mentioned reasons, we proposed a task for EVALITA 2016 concerning the domain adaptation of PoS-taggers to Twitter texts. Participants to the evaluation campaign were required to use the two following data sets provided by the organization to set up their systems: the first one, henceforth referred to as Development Set (DS), contains data manually annotated using a specific tagset (see section 2.2 for the tagset description) and must be used to train participants systems; the second one, referred to as Test Set

(TS), contains the test data in blind format for the evaluation and has been given to participants in the date scheduled for the evaluation.
For better focusing the task on the challenges related to PoS tagging, but also for avoiding the boring problem of disappeared tweets, the distributed version of tweets has been previously tokenised, splitting each token on a different line.
Moreover, according to an "open task" perspective, participants were allowed to use other resources with respect to those released for the task, both for training and to enhance final performances, as long as their results apply the proposed tagsets.

The paper is organized as follows. The next section describes the data exploited in the task, the annotation process and the issues related to the tokenisation and tagging applied to the dataset. The following section is instead devoted to the description of the evaluation metrics and participants results. Finally, we discuss the main issues involved in PoSTWITA.

2 Data Description

For the corpus of the proposed task, we collected tweets being part of the EVALITA2014 SENTIment POLarity Classification (SENTIPOLC) (Basile *et al.*, 2014) task dataset, benefitting of the fact that it is cleaned from repetitions and other possible sources of noise. The SENTIPOLC corpus originates from a set of tweets (Twita) randomly collected (Basile *et al.*, 2013), and a set of posts extracted exploiting specific keywords and hashtags marking political topics (SentiTUT) (Bosco *et al.*, 2013).

In order to work in a perspective of the development of a benchmark where a full pipeline of NLP tools can be applied and tested in the future, the same selection of tweets has been exploited in other EVALITA2016 tasks, in particular in the EVALITA 2016 SENTiment POLarity Classification Task (SENTIPOLC) (Barbieri *et al.*, 2016), Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) (Basile *et al.*, 2016) and Event Factuality Annotation Task (FactA) (Minard *et al.*, 2016).
Both the development and test set of EVALITA2016 has been manually annotated with PoS tags. The former, which has been distributed as the DS for PoSTWITA, includes 6,438 tweets (114,967 tokens). The latter, that is the TS, is instead composed by 300 tweets (4,759 tokens).

The tokenisation and annotation of all data have been first carried out by automatic tools, with a high error rate which is motivated by the features of the domain and text genre. We adapted the Tweet-NLP tokeniser (Gimpel *et al.*, 2011) to Italian for token segmentation and used the TnT tagger (Brants, 2000) trained on the Universal Dependencies corpus (v1.3) for the first PoS-tagging step (see also section 2.2).

The necessary manual correction has been applied by two different skilled humans working independently on data. The versions produced by them have been compared in order to detect disagreements, conflicts or residual errors which have been finally resolved by the contribution of a third annotator.
Nevertheless, assuming that the datasets of PoSTWITA are developed from scratch for what concerns the tokenisation and annotation of grammatical categories, we expected the possible presence of a few residual errors also after the above described three phases of the annotation process. Therefore, during the evaluation campaign, and before the date scheduled for the evaluation, all participants were invited and encouraged to communicate to the organizers any errors found in the DS. This allowed the organizers (but not the participants) to update and redistribute it to the participants in an enhanced form.

No lexical resource has been distributed with PoSTWITA 2016 data, since each participant is allowed to use any available lexical resource or can freely induce it from the training data.

All the data are provided as plain text files in UNIX format (thus attention must be paid to newline character format), tokenised as described in section 2.1, but only those of the DS have been released with the adequate PoS tags described in section 2.2. The TS contains only the tokenised words but not the correct tags, that have to be added by the participant systems to be submitted for the evaluation. The correct tokenised and tagged data of the TS (called gold standard TS), exploited for the evaluation, has been provided to the participants after the end of the contest, together with their score.

According to the treatment in the dataset from where our data are extracted, each tweet in PoSTWITA corpus is considered as a separate entity and

we did not preserved thread integrity, thus taggers participating to the contest have to process each tweet separately.

2.1 Tokenisation Issues

The problem of text segmentation (tokenisation) is a central issue in PoS-tagger evaluation and comparison. In principle, for practical applications, every system should apply different tokenisation rules leading to different outputs.

We provided in the evaluation campaign all the development and test data in tokenised format, one token per line followed by its tag (when applicable), following the schema:

```
_____ID_TWEET_1_____      _____162545185920778240_____
<TOKEN_1> <TAG1>          Governo PROPN
<TOKEN_2> <TAG2>          Monti PROPN
<TOKEN_3> <TAG3>          :  PUNCT
<TOKEN_4> <TAG4>          decreto NOUN
<TOKEN_5> <TAG5>          in ADP
<TOKEN_6> <TAG6>          cdm PROPN
<TOKEN_7> <TAG7>          per ADP
<TOKEN_8> <TAG8>          approvazione NOUN
<TOKEN_9> <TAG9>          !  PUNCT
<TOKEN_10> <TAG10>        http://t.co/Z76KLLGP URL

_____ID_TWEET_2_____      _____192902763032743936_____
<TOKEN_1> <TAG1>          #Ferrara HASHTAG
<TOKEN_2> <TAG2>          critica VERB
<TOKEN_3> <TAG3>          #Grillo HASHTAG
<TOKEN_4> <TAG4>          perché SCONJ
<TOKEN_n> <TAGn>          ...
```

The first line for each tweet contains the Tweet ID, while the line of each tweet after the last one is empty, in order to separate each post from the following. The example above shows some tokenisation and formatting issues, in particular:

- accents, which are coded using UTF-8 encoding table;
- apostrophe, which is tokenised separately only when used as quotation mark, not when signalling a removed character (like in *dell'/orto*)

All the other features of data annotation are described in details in the following parts of this section.

For what concerns tokenisation and tagging principles in EVALITA2016 PoSTWITA, we decided to follow the strategy proposed in the Universal Dependencies (UD) project for Italian[1] applying only minor changes, which are motivated by the special features of the domain addressed in the task. This makes the EVALITA2016-PoSTWITA gold standard annotation compliant with the other UD datasets, and strongly improves the portability of our newly developed datasets towards this standard.

Assuming, as usual and more suitable in PoS tagging, a neutral perspective with respect to the solution of parsing problems (more relevant in building treebanks), we differentiated our format from that one applied in UD, by maintaining the word unsplitted rather than splitted in different tokens, also in the two following cases:

- for the articulated prepositions (e.g. *dalla* (from-the[fem]), *nell´* (in-the[masc]), *al* (to-the), ...)
- for the clitic clusters, which can be attached to the end of a verb form (e.g. *regalaglielo* (gift-to-him-it), *dandolo* (giving-it), ...)

For this reason, we decided also to define two novel specific tags to be assigned in these cases (see section 1): *ADP_A* and *VERB_CLIT* respectively for articulated prepositions and clitics, according to the strategy assumed in previous EVALITA PoS tagging evaluations.

The participants are requested to return the test file using exactly the same tokenisation format, containing exactly the same number of tokens. The comparison with the reference file will be performed line-by-line, thus a misalignment will produce wrong results.

2.2 Tagset

Beyond the introduction of the novel labels cited above, motivated by tokenisation issues and related to articulated prepositions and clitic clusters, for what concerns PoS tagging labels, further modifications with respect to UD standard are instead motivated by the necessity of more specific labels to represent particular phenomena often occurring in social media texts. We introduced therefore new Twitter-specific tags for cases that following the UD specifications should be all classified into the generic *SYM* (symbol) class, namely emoticons, Internet addresses, email addresses, hashtags and mentions (*EMO*, *URL*, *EMAIL*, *HASHTAG* and *MENTION*). See Table 1 for a complete description of the PoSTWITA tagset.

We report in the following the more challenging issues addressed in the development of our data sets, i.e. the management of proper nouns and of foreign words.

[1] http://universaldependencies.org/it/pos/index.html

Tagset		Category	Examples
UD	**PoSTWITA16**		**if different from UD specs**
ADJ	ADJ	Adjective	-
ADP	ADP	Adposition (simple prep.)	di, a, da, in, con, su, per
	ADP_A	Adposition (prep.+Article)	dalla, nella, sulla, dell
ADV	ADV	Adverb	-
AUX	AUX	Auxiliary Verb	-
CONJ	CONJ	Coordinating Conjunction	-
DET	DET	Determiner	-
INTJ	INTJ	Interjection	-
NOUN	NOUN	Noun	-
NUM	NUM	Numeral	-
PART	PART	Particle	-
PRON	PRON	Pronoun	-
PROPN	PROPN	Proper Noun	-
PUNCT	PUNCT	punctuation	-
SCONJ	SCONJ	Subordinating Conjunction	-
SYM	SYM	Symbol	-
	EMO	Emoticon/Emoji	:-) ^_^ ♥ :P
	URL	Web Address	http://www.somewhere.it
	EMAIL	Email Address	someone@somewhere.com
	HASHTAG	Hashtag	#staisereno
	MENTION	Mention	@someone
VERB	VERB	Verb	-
	VERB_CLIT	Verb + Clitic pronoun cluster	mangiarlo, donarglielo
X	X	Other or RT/rt	-

Table 1: EVALITA2016 - PoSTWITA tagset.

2.2.1 Proper Noun Management

The annotation of named entities (NE) poses a number of relevant problems in tokenisation and PoS tagging. The most coherent way to handle such kind of phenomena is to consider each NE as a unique token assigning to it the PROPN tag. Unfortunately this is not a viable solution for this evaluation task, and, moreover, a lot of useful generalisation on n-gram sequences (e.g. *Ministero/dell/Interno* PROPN/ADP_A/PROPN) would be lost if adopting such kind of solution. Anyway, the annotation of sequences like *Banca Popolare* and *Presidente della Repubblica Italiana* deserve some attention and a clear policy.
Following the approach applied in Evalita 2007 for the PoS tagging task, we annotate as PROPN those words of the NE which are marked by the uppercase letter, like in the following examples:

Banca PROPN	Presidente PROPN	Ordine PROPN
Popolare PROPN	della ADP_A	dei ADP_A
	Repubblica PROPN	Medici PROPN
	Italiana PROPN	

Nevertheless, in some other cases, the uppercase letter has not been considered enough to determine the introduction of a PROPN tag:
"...anche nei Paesi dove..., *"...in contraddizione con lo Stato sociale..."*.
This strategy is devoted to produce a data set that incorporates the speakers linguistic intuition about this kind of structures, regardless of the possibility of formalization of the involved knowledge in automatic processing.

2.2.2 Foreign words

Non-Italian words are annotated, when possible, following the same PoS tagging criteria adopted in UD guidelines for the referring language. For instance, *good-bye* is marked as an interjection with the label INTJ.

3 Evaluation Metrics

The evaluation is performed in a black box approach: only the systems output is evaluated. The evaluation metric will be based on a token-by-

Team ID	Team	Affiliations
EURAC	E.W. Stemle	Inst. for Specialised Commun. and Multilingualism, EURAC Research, Bolzano/Bozen, Italy
ILABS	C. Aliprandi, L De Mattei	Integris Srl, Roma, Italy
ILC-CNR	A. Cimino, F. Dell'Orletta	Istituto di Linguistica Computazionale Antonio Zampolli CNR, Pisa, Italy
MIVOQ	Giulio Paci	Mivoq Srl, Padova, Italy
NITMZ	P. Pakray, G. Majumder	Deptt. of Computer Science & Engg., Nat. Inst. of Tech., Mizoram,Aizawl, India
UniBologna	F. Tamburini	FICLIT, University of Bologna, Italy
UniDuisburg	T. Horsmann, T. Zesch	Language Technology Lab Dept. of Comp. Science and Appl. Cog. Science, Univ. of Duisburg-Essen, Germany
UniGroningen	B. Plank, M. Nissim	University of Groningen, The Nederlands
UniPisa	G. Attardi, M. Simi	Dipartimento di Informatica, Universit di Pisa, Italy

Table 2: Teams participating at the EVALITA2016 - PoSTWITA task.

token comparison and only a single tag is allowed for each token. The considered metric is the Tagging accuracy: it is defined as the number of correct PoS tag assignment divided by the total number of tokens in TS.

4 Teams and Results

16 teams registered for this task, but only 9 submitted a final run for the evaluation. Table 2 outlines participants' main data: 7 participant teams belong to universities or other research centres and the last 2 represent private companies working in the NLP and speech processing fields.

Table 3 describes the main features of the evaluated systems w.r.t. the core methods and the additional resources employed to develop the presented system.

In the Table 4 we report the final results of the PoSTWITA task of the EVALITA2016 evaluation campaign. In the submission of the result, we allow to submit a single "official" result and, optionally, one "unofficial" result ("UnOFF" in the table): UniBologna, UniGroningen, UnPisa and UniDuisburg decided to submit one more unofficial result. The best result has been achieved by the ILC-CNR group (93.19% corresponding to $4,435$ correct tokens over $4,759$).

5 Discussion and Conclusions

Looking at the results we can draw some provisional conclusions about the PoS-tagging of Italian tweets:

- as expected, the performances of the automatic PoS-taggers when annotating tweets are lower than when working on normal texts, but are in line with the state-of-the art for other languages;
- all the top-performing systems are based on Deep Neural Networks and, in particular, on Long Short-Term Memories (LSTM) (Hochreiter, Schmidhuber, 1997; Graves, Schmidhuber, 1997);
- most systems use word or character embeddings as inputs for their systems;
- more or less all the presented systems make use of additional resources or knowledge (morphological analyser, additional tagged corpora and/or large non-annotated twitter corpora).

Looking at the official results, and comparing them with the experiments that the participants devised to set up their own system (not reported here, please look at the participants' reports), it is possible to note the large difference in performances. During the setup phase most systems, among the top-performing ones, obtained coherent results well above 95/96% of accuracy on the development set (either splitting it into a training/validation pair or by making cross-validation tests), while the best performing system in the official evaluation exhibit performances slightly above 93%. It is a huge difference for this kind of task, rarely observed in literature.

One possible reason that could explain this difference in performances regards the kind of docu-

Team ID	Core methods	Resources (other than DS)
EURAC	LSTM NN (word&char embeddings)	DiDi-IT
ILABS	Perceptron algorithm	word features extracted from proprietary resources and 250k entries of wikitionary.
ILC-CNR	two-branch BiLSTM NN (word&char embeddings)	Morhological Analyser (65,500 lemmas) + ItWaK corpus
MIVOQ	Tagger combination based on Yamcha	Evalita2009 Pos-tagged data ISTC pronunciation dictionary
NITMZ	HMM bigram model	-
UniBologna	Stacked BiLSTM NN + CRF (augmented word embeddings)	Morhological Analyser (110,000 lemmas) + 200Mw twitter corpus
UniDuisburg	CRF classifier	400Mw Twitter corpus
UniGroningen	BiLSTM NN (word embedding)	Universal Dependencies v1.3 74 kw tagged Facebook corpus
UniPisa	BiLSTM NN + CRF (word&char embeddings)	423Kw tagged Mixed corpus 141Mw Twitter corpus

Table 3: Systems description.

#	Team ID	Tagging Accuracy
1	ILC-CNR	0.9319 (4435)
2	UniDuisburg	0.9286 (4419)
3	UniBologna_UnOFF	0.9279 (4416)
4	MIVOQ	0.9271 (4412)
5	UniBologna	0.9246 (4400)
6	UniGroningen	0.9225 (4390)
7	UniGroningen_UnOFF	0.9185 (4371)
8	UniPisa	0.9157 (4358)
9	UniPisa_UnOFF	0.9153 (4356)
10	ILABS	0.8790 (4183)
11	NITMZ	0.8596 (4091)
12	UniDuisburg_UnOFF	0.8178 (3892)
13	EURAC	0.7600 (3617)

Table 4: EVALITA2016 - PoSTWITA participants' results with respect to Tagging Accuracy. "UnOFF" marks unofficial results.

ments in the test set. We inherited the development set from the SENTIPOLC task at EVALITA2014 and the test set from SENTIPOLC2016 and, maybe, the two corpora, developed in different epochs and using different criteria, could contain also different kind of documents. Differences in the lexicon, genre, etc. could have affected the training phase of taggers leading to lower results in the evaluation phase.

References

Barbieri, F., Basile, V., Croce, D., Nissim, M., Novielli, N., Patti, V. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. author=, *In Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*.

Basile, v., Bolioli, A., Nissim, M., Patti, V., Rosso, P. 2014. Overview of the Evalita 2014 SENTIment POLarity Classification Task *In Proceedings of Evalita 2014*, 50–57.

Basile, P., Caputo, A., Gentile, A.L., Rizzo, G. Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) Task. *In Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*.

Basile, V., Nissim, M. Sentiment analysis on Italian tweets. *In Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*.

Bosco, c., Patti, V., Bolioli, A. Developing Corpora for Sentiment Analysis: The Case of Irony and Senti-TUT. *IEEE Intelligent Systems, special issue on Knowledge-based approaches to content-level sentiment analysis*. Vol 28 num 2.

Brants, T. 2000. TnT – A Statistical Part-of-Speech Tagger. *In Proceedings of the 6th Applied Natural Language Processing Conference*.

Derczynski, L.,Ritter, A., Clark, S., Bontcheva, K. 2013. Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data *In Proceedings of RANLP 2013*, 198–206.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., Smith, N.A. 2011. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. *In Proceedings of ACL 2011*.

Graves, A, Schmidhuber, J. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5-6), 602–610.

Hochreiter, S., Schmidhuber, J. 1997. Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

Minard, A.L., Speranza, M., Caselli, T. 2016 The EVALITA 2016 Event Factuality Annotation Task (FactA). *In Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*.

Neunerdt, M., Trevisan, B., Reyer, M., Mathar, R. 2013. Part-of-speech tagging for social media texts. *Language Processing and Knowledge in the Web*. Springer, 139–150.

Owoputi, O., OConnor, B., Dyer, C., Gimpel, K., Schneider, N., Smith, N.A. 2013. Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters. *In Proceedings of NAACL 2013*.

Character Embeddings PoS Tagger vs HMM Tagger for Tweets

Giuseppe Attardi, Maria Simi
Dipartimento di Informatica
Università di Pisa
Largo B. Pontecorvo, 3
I-56127 Pisa, Italy
{attardi, simi}@di.unipi.it

Abstract

English. The paper describes our submissions to the task on PoS tagging for Italian Social Media Texts (PoSTWITA) at Evalita 2016. We compared two approaches: a traditional HMM trigram Pos tagger and a Deep Learning PoS tagger using both character-level and word-level embeddings. The character-level embeddings performed better proving that they can provide a finer representation of words that allows coping with the idiosyncrasies and irregularities of the language in microposts.

Italiano. *Questo articolo descrive la nostra partecipazione al task di PoS tagging for Italian Social Media Texts (PoSTWITA) di Evalita 2016. Abbiamo confrontato due approcci: un PoS tagger tradizionale basato su HMM a trigrammi e un PoS Tagger con Deep Learning che usa embeddings sia a livello di caratteri che di parole. Gli embedding a caratteri hanno fornito un miglior risultato, dimostrando che riescono a fornire una rappresentazione più fine delle parole che consente di trattare le idiosincrasie e irregolarità del linguaggio usato nei micropost.*

1 Introduction

The PoS tagging challenge at Evalita 2016 was targeted to the analysis of Italian micropost language, in particular the language of Twitter posts. The organizers provided an annotated training corpus, obtained by annotating a collection of Italian tweets from the earlier Evalita 2014 SENTIPOLC corpus. The annotations follow the guidelines proposed by the Universal Dependencies (UD) project for Italian[1], in particular with respect to tokenization and tag set, with minor changes due to the specificity of the text genre. A few specific tags (EMO, URL, EMAIL, HASHTAG and MENTION), have been in fact added for typical morphological categories in social media texts, like emoticons and emoji's, web URL, email addresses, hashtags and mentions.

The challenge for PoS tagging of microposts consists in dealing with misspelled, colloquial or broken words as well as in overcoming the lack of context and proper uppercasing, which provide helpful hints when analysing more standard texts.

We conducted preparatory work that consisted in customizing some available lexical and training resources for the task: in section 2 and 3 we will describe such a process.

We decided to address the research question of comparing the relative performance of two different approaches to PoS tagging: the traditional word-based approach, based on a Hidden Markov Model PoS tagger, with a Deep Learning approach that exploits character-level embeddings (Ma and Hovy, 2016). Section 4 and 5 describe the two approaches in detail.

2 Building a larger training resource

The gold training set provided for the task consists in a collection of 6,640 Italian tweets from the Evalita 2014 SENTIPOLC corpus (corresponding to 127,843 word tokens). Given the relative small size of the resource, we extended it by leveraging on existing resources. We used the corpus previously used in the organization of the Evalita 2009 task on PoS Tagging (Attardi and

[1] http://universaldependencies.org/it/pos/index.html

Simi 2009), consisting in articles from the newspaper "La Repubblica", some articles from the Italian Wikipedia, and portions of the Universal Dependencies Italian corpus and a small collection of annotated Italian tweets. Table 1 provides details of the composition of the training resource.

Resource	Number of tokens
repubblica.pos	112,593
extra.pos	130
quest.pos	9,826
isst_tanl.pos	80,794
tut.pos	97,558
it-twitter.pos	1,018
Evalita 2016	*121,405*
Total	**423,324**

Table 1. Composition of the training set.

The tag set was converted to the Universal Dependencies schema taking into account the variants introduced in the task (different tokenization of articulated prepositions and introduction of ADP_A).

During development, the gold dataset provided by the organizers was split into two parts: a subset of about 105,300 tokens was used for training, while the remaining tokens were used as validation set (~22,500 tokens).

3 Normalization of URLs, emoticons and emoji's

In order to facilitate the tagging of morphological categories specifically introduced for social media texts, we applied a pre-processing step for normalizing the word forms. This was done by means of a set of rewriting rules based on regular expressions.

These rules are quite straightforward for URLs, hashtags, emails and mentions, while the identification of emoticons and emoji's required a set of carefully handcrafted rules because of their variety and higher degree of ambiguity.

4 The traditional approach: the TANL tagger

Linear statistical models, such as Hidden Markov Models (HMM) or Conditional Random Fields (CRF) are often used for sequence labeling (PoS tagging and NER).

In our first experiment, we used the Tanl Pos Tagger, based on a second order HMM.

The Tanl PoS tagger is derived from a rewriting in C++ of HunPos (Halácsy, et al. 2007), an open source trigram tagger, written in OCaml. The tagger estimates the probability of a sequence of labels $t_1 \ldots t_T$ for a sequence of words $w_1 \ldots w_T$ from the probabilities of trigrams:

$$\underset{t_1 \ldots t_T}{\operatorname{argmax}} P(t_{T+1}|t_T) \prod_{i=1}^{T} P(t_i|t_{i-1}, t_{i-2}) P(w_i|t_{i-1}, t_i)$$

The trigram probabilities are estimated smoothing by linear interpolation the probabilities of unigrams, bigrams and trigrams:

$$P(t_3|t_1, t_2) = \lambda_1 \hat{P}(t_3) \lambda_2 \hat{P}(t_3|t_2) \lambda_3 \hat{P}(t_3|t_1 t_2)$$

where $\hat{P}$ are maximum likelihood estimates and $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

An approximate Viterbi algorithm is used for finding the sequence of tags with highest probability, which exploit beam search to prune unlikely alternative paths.

The tagger uses a suffix guessing algorithm for dealing with unseen words. The tagger computes the probability distribution of tags for each suffix, by building a trie from the suffixes, up to a maximum length (default 10), of words appearing less than *n* (default 10) times in the training corpus. Actually two suffix tries are built: one for words beginning with uppercase, one for lowercase words. A word at the beginning of a sentence is looked up in its lowercase variant.

Special handling is provided for numbers and HTML entities.

The tagger can also be given a file with a list of possible tags and lemmas for each word, in order to initialize its lexicon. In our experiments we used a lexicon of 130 thousands Italian words.

5 Character-level Embeddings

Traditional techniques of statistical machine learning usually require, to perform best, task specific selection and tuning of hand-crafted features as well as resources like lexicons or gazetteers, which are costly to develop.

Recently, end-to-end approaches based on Deep Learning architectures have proved to be equally effective, without the use of handcrafted features or any data pre-processing, exploiting word embeddings as only features.

In order to deal with sequences, Collobert et al. (2011) proposed a Convolutional Neural Networks (CNN), trained to maximize the overall sentence level log-likehood of tag sequences, which was able to achieve state of the art accura-

cy on English PoS tagging. More recently, Recursive Neural Networks (RNN) have been proposed.

The word embeddings exploited as features in these systems proved suitable to represent words in well formed texts like the news articles used in the CoNNL PoS tagging benchmarks.

We conjectured that dealing with the noisy and malformed texts in microposts might require features at a finer level than words, i.e. to use character-level embeddings. Hence we devised an experiment to explore the effectiveness of combining both character-level and word-level embeddings in PoS tagging of tweets.

We based our experiments on the work by Ma and Hovy (2016), who propose an approach to sequence labeling using a bi-directional long-short term memory (BiLSTM) neural network, a variant of RNN. On top of the BiLSTM, a sequential CRF layer can be used to jointly decode labels for the whole sentence.

The implementation of the BiLSTM network is done in Lasagne[2], a lightweight library for building and training neural networks in Theano[3].

For training the BiLSTM tagger we used word embeddings for tweets created using the `fastText` utility[4] (Bojanowski et al., 2016) on a collection of 141 million Italian tweets retrieved over the period from May to September 2016 using the Twitter API. Selection of Italian tweets was achieved by using a query containing a list of the 200 most common Italian words.

The embeddings were created with dimension 100, using a window of 5 and retaining words with a minimum count of 100, for a total of 245 thousands words.

6 Results

The following table reports the top 9 official scores obtained by participant systems.

Submission	Accuracy	Correct
Team1	0.9319	4435
Team2	0.9285	4419
Team3_UNOFFICIAL	0.9279	4416
Team4	0.9270	4412
Team3	0.9245	4400
Team5	0.9224	4390
Team5_UNOFFICIAL	0.9184	4371
UNIPI	**0.9157**	**4358**
UNIPI_UNOFFICIAL	**0.9153**	**4356**

Table 2. PoSTWITA top official results.

After submission we performed another experiment with the BiLSTM tagger, increasing the dimension of word embeddings from 100 to 200 and obtained an accuracy of 92.50% (4402/4759).

To further test the ability of the character-level embeddings to deal completely autonomously with the original writings of tweets, we performed a further experiment where we supply the original text of tweets without normalization. This experiment achieved an accuracy of 91.87% (4372/4759), proving that indeed the RNN character-level approach is capable of learning by itself even unusual tokens, recognizing quite well also emoticons and emoji's, without any need of preconceived linguistic knowledge, encoded in an ad-hoc rule system.

7 Discussion

While the results with the two approaches, used in the official and unofficial run, are strikingly close (a difference of only two errors), the two taggers differ significantly on the type of errors they make.

7.1 Error analysis

Table 3 reports a breakdown of the errors over PoS categories, for both systems, in order to appreciate the difference in behaviour. Note that a single PoS mismatch is counted twice, once for each PoS involved. Three cases of misspelled PoS in the gold test were corrected before this analysis.

	BiLSTM	HMM
URL	5	2
EMO	**36**	**6**
DET	32	37
AUX	27	19
CONJ	5	2
NOUN	**132**	**155**
PUNCT	8	5
MENTION	1	0
NUM	16	14
ADP_A	8	7
ADV	44	51
VERB_CLIT	4	3
ADP	26	27
SCONJ	**15**	**26**

[2] https://github.com/Lasagne
[3] https://github.com/Theano/Theano
[4] https://github.com/facebookresearch/fastText.git

PROPN	**136**	**150**
INTJ	**44**	**34**
VERB	**110**	**83**
X	34	31
ADJ	**67**	**86**
SYM	3	5
PRON	**42**	**56**
HASHTAG	1	1
TOTAL	**796**	**800**

Table 3. Breakdown of errors over PoS types.

As previously mentioned, social media specific tags are not the most difficult problem. To be fair, we noticed that the official BiLSTM run is plagued by a suspicious high number of errors in identifying EMO's. However, by checking the steps in the experiment, we discovered that this poor performance was due to a mistake in the normalization step.

Confusion between NOUN and PROPN represents the largest source of errors. In the official run there are 66 errors (35 PROPN tagged as NOUN, 33 NOUN tagged as PROPN), corresponding to nearly 17% of all the errors. The traditional unofficial run does even worse: 19% of the errors are due to this confusion.

Both taggers are weak in dealing with improper use of case (lower case proper names and all caps texts), which is very common in Twitter posts. This could be because the training set is still dominated by more regular texts where the case is a strong indication of proper names. In addition, the annotation style chosen for long titles, not fully compliant with UD, makes the task even more difficult. For example the event "Settimana della moda femminile/*Women fashion week*" or "Giornata mondiale vittime dell'amianto/*World Day of the victims of the asbestos*" are annotated as a sequence of PROPN in the gold test set as opposed to using the normal grammatical conventions, as specified in the UD guidelines.

The traditional system is slightly more accurate in predicting the distinction between VERB (main verbs) and AUX (auxiliary and modal verbs): 19 errors against 26.

8 Conclusions

We explored using both a traditional HMM trigram PoS tagger and a Deep Learning PoS Tagger that uses both character and word-level embeddings, in the analysis of Italian tweets.

The latter tagger uses embeddings as only features and no lexicon nor other linguistic resource. The tagger performs surprisingly well, with an unofficial run that ranks among the top 5. This confirms our conjecture that character-level embeddings are able of coping with the idiosyncrasies and irregular writings in microposts.

Acknowledgments

We gratefully acknowledge the support by the University of Pisa through project PRA and by NVIDIA Corporation through a donation of a Tesla K40 GPU used in the experiments.

References

Giuseppe Attardi and Maria Simi. 2009. Overview of the EVALITA. Part-of-Speech Tagging Task. *Proceedings of Workshop Evalita 2009*, Reggio Emilia.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. https://arxiv.org/abs/1607.04606

Ronan Collobert, JasonWeston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research 12. 2461-2505.

Péter Halácsy, András Kornai and Csaba Oravecz. 2007. HunPos – an open source trigram tagger. *Proceedings of the Demo and Poster Sessions of the 54th Annual Meeting of the ACL*, pp. 209-212.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (ACL 2016), pp. 1064-1074, Berlin, Germany. August 2016.

Building the state-of-the-art in POS tagging of Italian Tweets

Andrea Cimino and **Felice Dell'Orletta**
Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC–CNR)
ItaliaNLP Lab - *www.italianlp.it*
{andrea.cimino, felice.dellorletta}@ilc.cnr.it

Abstract

English. In this paper we describe our approach to EVALITA 2016 POS tagging for Italian Social Media Texts (PoSTWITA). We developed a two-branch bidirectional Long Short Term Memory recurrent neural network, where the first bi-LSTM uses a typical vector representation for the input words, while the second one uses a newly introduced word-vector representation able to encode information about the characters in the words avoiding the increasing of computational costs due to the hierarchical LSTM introduced by the character–based LSTM architectures. The vector representations calculated by the two LSTM are then merged by the sum operation. Even if participants were allowed to use other annotated resources in their systems, we used only the distributed data set to train our system. When evaluated on the official test set, our system outperformed all the other systems achieving the highest accuracy score in EVALITA 2016 PoSTWITA, with a tagging accuracy of 93.19%. Further experiments carried out after the official evaluation period allowed us to develop a system able to achieve a higher accuracy. These experiments showed the central role played by the handcrafted features even when machine learning algorithms based on neural networks are used.

Italiano. *In questo articolo descriviamo il sistema che abbiamo utilizzato per partecipare al task POS tagging for Italian Social Media Texts (PoSTWITA) della conferenza EVALITA 2016. Per questa partecipazione abbiamo sviluppato un sistema basato su due reti neurali parallele entrambi bidirezionali e ricorrenti di tipo Long Short Term Memory (LSTM). Mentre la prima rete neurale è una LSTM bidirezionale che prende in input vettori che rappresentano le parole in maniera tipica rispetto a precedenti lavori, la seconda prende in input una nuova rappresentazione vettoriale delle parole che contiene informazioni sui caratteri contenuti evitando un incremento del costo computazionale del sistema rispetto a LSTM che prendono in input rappresentazioni vettoriali delle sequenze di caratteri. Le rappresentazioni vettoriali ottenute dalle due LSTM vengono in fine combinate attraverso l'operatore di somma. Il nostro sistema, utilizzando come dati annotati solo quelli distribuiti dagli organizzatori del task, quando valutato sul test set uffciale ha ottenuto il miglior risultato nella competizione EVALITA 2016 PoSTWITA, riportando una accuratezza di 93.19%. Ulteriori esperimenti condotti dopo il periodo ufficiale di valutazione ci hanno permesso di sviluppare un sistema capace di raggiungre una accuratezza ancora maggiore, mostrandoci l'importanza dell'ingegnerizzazione manuale delle features anche quando vengono utilizzati algoritmi di apprendimento basati su reti neurali.*

1 Description of the system

Our approach to EVALITA 2016 PoSTWITA (Bosco et al., 2016) task was implemented in a software prototype operating on tokenized sentences which assigns to each token a score expressing its probability of belonging to a given part-of-speech class. The highest score represents the most probable class.

Differently from the previous EVALITA part of speech tagging tasks (Tamburini (2007), Attardi and Simi (2009)), in EVALITA 2016 PoSTWITA the participants must tackle the problem of analyzing text with low conformance to common writing practices. For example, capitalization rules may be ignored; excessive punctuation, particularly repeated ellipsis and question marks may be used, or spacing may be irregular (Agichtein et al., 2008). Our development system strategy took into account this issue. In particular, we implemented a multiple input bidirectional Long Short Term Memory recurrent neural network (LSTM) model. We developed a two-branched bidirectional LSTM (bi-LSTM) where the first bi-LSTM uses a typical vector representation of the input words commonly used for different classification tasks, while the second one uses a newly introduced word-vector representation specifically designed to handle peculiarities of ill-formed or not standard texts typical of social media texts.

To create the input vectors for the two branches we use a combination of different components extracted from three different word embedding lexicons, from a manually created morpho-syntactic lexicon and from handcrafted features specifically defined to improve the accuracy of the system when tested on social media texts.

In this work we used Keras (Chollet, 2016) deep learning framework to generate the neural network models.

1.1 Lexicons

In order to improve the overall accuracy of our system, we developed three word embedding lexicons[1] and we used a manually created morpho-syntactic lexicon.

1.1.1 Word Embedding lexicons

Since the lexical information in tweets can be very sparse, to overcame this problem we built three word embedding lexicons.

For this purpose, we trained two predict models using the word2vec[2] toolkit (Mikolov et al., 2013). As recommended in (Mikolov et al., 2013), we used the CBOW model that learns to predict the word in the middle of a symmetric window based on the sum of the vector representations of the words in the window. For our experiments, we considered a context window of 5 words. These models learn lower-dimensional word embeddings. Embeddings are represented by a set of latent (hidden) variables, and each word is a multidimensional vector that represent a specific instantiation of these variables. We built two Word Embedding Lexicons starting from the following corpora:

- The first lexicon was built using a tokenized version of the itWaC corpus[3]. The itWaC corpus is a 2 billion word corpus constructed from the Web limiting the crawl to the *.it* domain and using medium-frequency words from the Repubblica corpus and basic Italian vocabulary lists as seeds.
- The second lexicon was built from a tokenized corpus of tweets. This corpus was collected using the Twitter APIs and is made up of 10,700,781 Italian tweets.

In addition to these two lexicons, we built another word embedding lexicon based on fastText (Bojanowski et al., 2016), a library for efficient learning of word representations and sentence classification. FastText allows to overcome the problem of out-of-vocabulary words which affects the relying methodology of word2vec. Generating out-of-vocabulary word embeddings is a typical issue for morphologically rich languages with large vocabularies and many rare words. FastText overcomes this limitation by representing each word as a bag of character n-grams. A vector representation is associated to each character n-gram and the word is represented as the sum of these character n-gram representations. To build the lexicon based on fastText, we adopted as learning corpus the same set of tokenized tweets used to build the word2vec based lexicon.

1.1.2 Morpho-syntactic lexicon

We used a large Italian lexicon of about 1,300,000 forms, developed as part of the SemaWiki project[4]. The full-form lexicon was generated from a base lexicon of 65,500 lemmas, initially inspired by the Zanichelli dictionary[5], and updated along several years and cross-checked with other online dictionaries[6]. For each form the lexicon

[1] The three word embedding lexicons are freely available at the following website: *http://www.italianlp.it/*.

[2] http://code.google.com/p/word2vec/

[3] http://wacky.sslmit.unibo.it/doku.php?id=corpora

[4] http://medialab.di.unipi.it/wiki/SemaWiki

[5] Zingarelli: Il nuovo Zingarelli minore, 2008.

[6] Aldo Gabrielli: Il Grande Dizionario di Italiano; Tullio De Mauro: Il Dizionario della lingua italiana.

contains all the possible parts-of-speech and provides information on morpho-syntactic features, but using a different tagset (*ISST-TANL Tagsets*[7]) with respect to the one used for PoSTWITA.

1.2 The POS tagger architecture

The LSTM unit was initially proposed by Hochreiter and Schmidhuber (Hochreiter et al., 1997). LSTM units are able to propagate an important feature that came early in the input sequence over a long distance, thus capturing potential long-distance dependencies. This type of neural network was recently tested on Sentiment Analysis tasks (Tang et al., 2015), (Xu et al., 2016) where it has been proven to outperform classification performance in several sentiment analysis task (Nakov et al., 2016) with respect to commonly used learning algorithms, showing a 3-4 points of improvements. Similar big improvements have not been obtained in tagging tasks, such as Part-Of-Speech tagging. This is most due to the fact that state-of-the art systems for part of speech tagging exploit strong performing learning algorithms and hard feature engineering. In addition, a little knowledge of the surrounding context is enough to reach very high tagging performance. On the contrary, LSTM networks perform very well with respect to other learning algorithms when word dependencies are long. Although without a big improvement, POS tagging systems which exploit LSTM as learning algorithm have been proven to reach state-of-the-art performances both when analyzing text at character level (Ling et al., 2015) and at word level (Wang et al., 2016). More specifically they used a bidirectional LSTM allows to capture long-range dependencies from both directions of a sentence by constructing bidirectional links in the network (Schuster et al., 1997). In addition, (Plank et al., 2016) have proposed a model which takes into account at the same time both word level and character level information, showing very good results for many languages. As proposed by these systems, we employed a bidirectional LSTM architecture. We implemented a 2-branch bidirectional LSTM but instead of using the character based branch we introduced another specific word level branch in order to reduce the computational cost of the hierarchical LSTM introduced by the character based LSTM. This branch encodes information about the characters in each word of a sentence. The vector representations calculated by the two LSTM are then merged by the sum operation. For what concerns the optimization process, categorical cross-entropy is used as a loss function and the optimization process is performed by the rmsprop optimizer (Tieleman and Hinton, 2012). Each bidirectional LSTM branch is configured to have 24 units. In addition, we applied a dropout factor to both input gates and to the recurrent connections in order to prevent overfitting which is a typical issue of neural networks (Galp and Ghahramani, 2015). As suggested in (Galp and Ghahramani, 2015) we have chosen a dropout factor value in the optimum range $[0.3, 0.5]$, more specifically 0.35 for each branch.

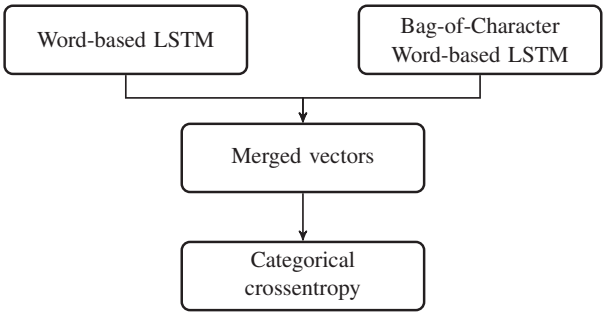


Figure 1: Diagram of the two-branched bi-LSTM architecture.

1.2.1 Word-based bi-LSTM

In this part, we describe the Word-based bidirectional LSTM branch of the proposed neural network architecture and the word level information given in input to this layer. Each word is represented by a low dimensional, continuous and real-valued vector, also known as word embedding and all the word vectors are stacked in a word embedding matrix. To train this LSTM branch, each input word in the tweet is represented by a 979-dimensional vector which is composed by:

Word2vec word embeddings: the concatenation of the two word embeddings extracted by the two available *word2vec* Word Embedding Lexicons (128 components for each word embedding, thus resulting in a total of 256 components), and for each word embedding an extra component was added in order to handle the "unknown word" (2 components).

FastText word embeddings: the word embeddings extracted by the *fastText* Word Embedding Lexicon (128 components).

[7] http://www.italianlp.it/docs/ISST-TANL-POStagset.pdf

Morpho-syntactic category: the parts-of-speech and the corresponding morpho-syntactic features obtained by exploiting the Morpho-syntactic lexicon, resulting in 293 components.
Spell checker: the parts-of-speech and the corresponding morpho-syntactic features of the word obtained by analyzing the current word using a spell checker (*pyenchant*[8]) and exploiting the Morpho-syntactic lexicon, resulting in 295 components.
Word length: a component representing the length of the analyzed word.
Is URL: a component indicating whether the "http" substring is contained in the analyzed word.
Is uppercase: a component indicating if the analyzed word is uppercase.
Is capitalized: a component indicating if the analyzed word is capitalized.
End of sentence: a component indicating whether or not the sentence was totally read.

1.2.2 Bag-of-Character Word-based bi-LSTM

In this part, we describe the Bag-of-Character Word-based bidirectional LSTM branch of the proposed neural network architecture and the word level information given in input to this layer. Differently from the Word-based LSTM branch, in this branch we did not use pretrained vectors. To train this LSTM branch, each input word in the tweet is represented by a 316-dimensional vector which is composed by:
Characters: a vector representing the set of characters which compose the current word. Since our considered alphabet is composed by 173 different characters, the resulting in a 173-dimensional vector.
Lowercased characters: 134 components representing the set of lowercased characters which compose the current word.
Has numbers: a component indicating whether or not the current word contains a number.
Contains not numbers: a component indicating whether or not the current word contains non numbers.
Contains lowercased: a component indicating whether or not the current word contains lowercase characters.
Contains uppercased: a component indicating whether or not the current word contains uppercase characters.
Contains alphanumeric: a component indicating whether or not the current word contains alphanumeric characters
Contains not alphanumeric: a component indicating whether or not the current word contains non alphanumeric characters
Contains alphabetics: a component indicating whether or not the current word contains alphabetic characters.
Contains not alphabetics: a component indicating whether or not the current word contains non alphabetic characters.
End of sentence: a component indicating whether the sentence was totally read.

[8] http://pythonhosted.org/pyenchant/

2 Results and Discussion

To develop our system, we created an internal development set of 368 tweets randomly selected from the training set distributed by the task organizers. The first row in Table 1 reports the accuracy achieved by our final system on the internal development set and on the official test set (row *Two-branch bi-LSTM*).

Configuration	Devel	Test
Two-branch bi-LSTM	**96.55**	93.19
Word bi-LSTM	96.03	92.35
Bag-of-Char. Word bi-LSTM	84.47	80.77
No Morpho-syntactic lexicon	96.48	**93.54**
No spell checker	96.49	93.31
No word2vec lexicons	93.23	89.87
No fastText lexicon	95.85	92.43
No feature engineering	96.39	93.06

Table 1: Tagging accuracy (in percentage) of the different learning models on our development set and the official test set.

We tested different configurations of our system in order to evaluate the contribution on the tagging accuracy of: *i)* each branch in the proposed architecture, *ii)* the different word embedding and morpho-syntactic lexicons and *iii)* the handcrafted features. We carried out different experiments that reflect the questions we wanted to answer, more specifically the questions are:

- (a) what are the contributions of the *Word-based bi-LSTM* and of the *Bag-of-Character Word-based bi-LSTM*?

- (b) what is the contribution of the *Morpho-syntactic lexicon*?
- (c) what is the contribution of the spell checker?
- (d) what is the contribution of fastText with respect to word2vec *Word Embedding lexicons*?

In order to answer to the question (a), first we run the Word-based LSTM excluding the Bag-of-Character Word-based bi-LSTM branch, then we excluded the Word-based bi-LSTM to verify the Bag-of-Character Word based bi-LSTM contribution. The results of these experiments are reported in *Word bi-LSTM* and *Bag-of-Char. Word bi-LSTM* rows in Table 1. The Word-based bi-LSTM is clearly the best performer with respect to the Bag-of-Character one, but remarkable is that our proposed two-branch architecture shows an improvement of about 0.5 points in the development set with respect to the best single bi-LSTM. The same behaviour is shown in the test set, where the combined system achieves an improvement of 0.84 points with respect to the single Word-based bi-LSTM.

In order to answer to the question (b), we excluded from the input vectors of the Word-based bi-LSTM branch the morpho-syntactic category components extracted from Morpho-syntactic lexicon. Row *No Morpho-syntactic lexicon* reports the results and shows that this information gives a negligible improvement on the development set and unexpectedly a slight drop on the test set.

For what concerns the question (c), we excluded the morpho-syntactic category components of the word obtained using the spell checker. The results are reported in the *No spell checker* row. Similarly to what happened in the (b) experiment, also such information do not contribute in increasing the tagging performances.

In order to compare the contributions of fastText and word2vec lexicons (question (d)), we considered two different system configurations: one removing the two word2vec lexicons (*No word2vec lexicons* row) and one removing fastText and itWac word2vec lexicons (*No fastText lexicon* row). In this second configuration, we removed also the itWac word2vec lexicon to compare fastText and word2vec using the same learning corpus (the twitter corpus described in section 1.1.1). In both configurations we excluded the other Word-based LSTM components, while we left all the components of the Bag-of-Character Word-based LSTM. The results show that word2vec seems to be a better choice with respect to fastText, both in development and in test sets. This is in contrast with what we would have expected considering that fastText learns the word embedding representation using subword information that should be particularly useful for the analysis of non standard text such as social media ones.

2.1 Single bi-LSTM and Handcrafted features

After the submission of the final system results, we devised two further experiments. The first one was devoted to testing the tagging performances of a single word-based bi-LSTM architecture with respect to the presented Two-branch bi-LSTM. The second experiment was aimed to study the effect of handcrafted features combined with the learning ones. To this aim, we developed a Part-of-Speech tagger based on a single word-based bi-LSTM, where each input word vector is the concatenation of the two input word representations of the bi-LSTMs presented in Section 1.2.1 and Section 1.2.2.

Table 2 reports the results of these experiments. As shown in the *Single bi-LSTM* row, the use of the single architecture instead of the two-branch one does not affect tagging results, actually the single bi-LSTM slightly outperforms the two-branch architecture when tested on the test set (+0.48%).

In order to evaluate the effect of handcrafted features, we conducted a last experiment where we removed all the components from the input vectors of the single Word-based bi-LSTM with the exceptions of word2vec and fastText word embeddings. *No handcrafted features* row shows the relevance of the handcrafted features that yield an improvement of 1.34% and 1.68% on the development and the test sets respectively. These results show the important role of *feature engineering* even when neural networks learning algorithms are used.

3 Conclusion

In this paper we reported the results of our participation to the EVALITA 2016 POS tagging for Italian Social Media Texts (PoSTWITA). By resorting to a two-branch bidirectional LSTM, word em-

Configuration	Devel	Test
Single bi-LSTM	**96.39**	**93.67**
No handcrafted features	95.22	91.99

Table 2: Tagging accuracy of the single word-based bi-LSTM on our development set and the official test set.

beddings and morpho-syntactic lexicons and hand crafted features we achieved the best score. In particular, we showed the relevance of handcrafted features that allowed an improvement of more than one percentage point in terms of tagging accuracy both in development and test sets when combined with learned features such as word embedding lexicons. As future research direction we will test the contribution of a pure character based LSTM with respect to character handcrafted features.

References

Eugene Agichtein and Carlos Castillo and Debora Donato and Aristides Gionis and Gilad Mishne. 2008. Finding High-quality Content in Social Media. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*. New York, USA.

Giuseppe Attardi and Maria Simi. 2009. Overview of the EVALITA 2009 Part-of-Speech Tagging Task. In *Proceedings of Evalita '09, Evaluation of NLP and Speech Tools for Italian*. December, Reggio Emilia, Italy.

Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:607.04606*.

Cristina Bosco and Fabio Tamburini and Andrea Bolioli and Alessandro Mazzei. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task. In *Proceedings of Evalita '16, Evaluation of NLP and Speech Tools for Italian*. December, Naples, Italy.

François Chollet. 2016. Keras. *Software available at https://github.com/fchollet/keras/tree/master/keras*.

Cìcero Nogueira dos Santos and Bianca Zadrozny. 2013. Learning Character-level Representations for Part-of-Speech Tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*.

Yarin Gal and Zoubin Ghahramani. 2015. A theoretically grounded application of dropout in recurrent neural networks. *arXiv preprint arXiv:1512.05287*.

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

Wang Ling, Chris Dyer, Alan W. Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo and Luis Tiago 2016. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1520–1530 , Lisbon, Portugal. ACL.

Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv1:1301.3781.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani and Veselin Stoyanov 2016. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*.

Barbara Plank, Anders Søgaard and Yoav Goldberg. 2016. Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models an Auxiliary Loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. August, Berlin, Germany.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681.

Fabio Tamburini. 2007. Evalita 2007: The Part-of-Speech Tagging Task. In *Proceedings of Evalita '07, Evaluation of NLP and Speech Tools for Italian*. September, Rome, Italy.

Duyu Tang, Bing Qin and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* 1422-1432, Lisbon, Portugal.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-RmsProp: Divide the gradient by a running average of its recent magnitude. In *COURSERA: Neural Networks for Machine Learning*.

XingYi Xu, HuiZhi Liang and Timothy Baldwin. 2016. UNIMELB at SemEval-2016 Tasks 4A and 4B: An Ensemble of Neural Networks and a Word2Vec Based Model for Sentiment Classification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*.

Peilu Wang, Yao Qian, Frank K. Soong, Lei He, and Hai Zhao . 2016. Learning Distributed Word Representations For Bidirectional LSTM Recurrent Neural Network. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* 527–533, San Diego, CA, USA. ACL.

Building a Social Media Adapted PoS Tagger Using FlexTag – A Case Study on Italian Tweets

Tobias Horsmann **Torsten Zesch**
Language Technology Lab
Department of Computer Science and Applied Cognitive Science
University of Duisburg-Essen, Germany
{tobias.horsmann,torsten.zesch}@uni-due.de

Abstract

English. We present a detailed description of our submission to the PoSTWITA shared-task for PoS tagging of Italian social media text. We train a model based on FlexTag using only the provided training data and external resources like word clusters and a PoS dictionary which are build from publicly available Italian corpora. We find that this minimal adaptation strategy, which already worked well for German social media data, is also highly effective for Italian.

Italiano. *Vi presentiamo una descrizione dettagliata della nostra partecipazione al task di PoS tagging for Italian Social Media Texts (PoSTWITA). Abbiamo creato un modello basato su FlexTag utilizzando solo i dati forniti e alcune risorse esterne, come cluster di parole e un dizionario di PoS costruito da corpora italiani disponibili pubblicamente. Abbiamo scoperto che questa strategia di adattamento minimo, che ha già dato buoni risultati con i dati di social media in tedesco, è altamente efficace anche per l'Italiano.*

1 Introduction

In this paper, we describe our submission to the PoSTWITA Shared-Task 2016 that aims at building accurate PoS tagging models for Italian Twitter messages. We rely on FLEXTAG (Zesch and Horsmann, 2016), a flexible, general purpose PoS tagging architecture that can be easily adapted to new domains and languages. We re-use the configuration from Horsmann and Zesch (2015) that has been shown to be most effective for adapting a tagger to the social media domain. Besides training on the provided annotated data, it mainly relies on external resources like PoS dictionaries and word clusters that can be easily created from publicly available Italian corpora. The same configuration has been successfully applied for adapting FlexTag to German social media text (Horsmann and Zesch, 2016).

2 Experimental Setup

We use the FlexTag CRF classifier (Lafferty et al., 2001) using a context window of ± 1 tokens, the 750 most-frequent character ngrams over all bi, tri and four-grams and boolean features if a token contains a hyphen, period, comma, bracket, underscore, or number. We furthermore use boolean features for capturing whether a token is fully capitalized, a retweet, an url, a user mention, or a hashtag.

Data We train our tagging model only on the annotated data provided by the shared task organizers. As this training set is relatively large, we decided against adding additional annotated data from foreign domains which is a common strategy to offset small in-domain training sets (Ritter et al., 2011; Horsmann and Zesch, 2016).

Resources *Word clusters:* We create word clusters using Brown clustering (Brown et al., 1992) from 400 million tokens of Italian Twitter messages which have been crawled between the years 2011 and 2016.

PoS dictionary: We create a PoS dictionary which stores the three most frequent PoS tags of a word. We build the dictionary using a PoS annotated Italian Wikipedia corpus.[1]

Namelists: We furthermore use lists of first names obtained from Wikipedia and extract words tagged as named entities from the ItWaC web corpus (Baroni et al., 2009) to improve coverage of named entities.

[1] http://wacky.sslmit.unibo.it/doku.php?id=corpora

	Acc All	Acc OOV
TreeTagger Baseline	75.5	-
PoSTWITA	90.6	80.5
+ Clusters	92.7	85.6
+ PoS-Dict	92.2	85.3
+ Namelist	91.1	81.4
+ All Resources	**92.9**	**86.2**

Table 1: Results on the test data set

Baseline System We compare our results to the Italian model of TreeTagger (Schmid, 1995). As TreeTagger uses a much more fine-grained tagset than the one used in this shared-task, we map the fine tags mapping as provided by DKPro Core DKProCore (Eckart de Castilho and Gurevych, 2014).

3 Results

Table 1 gives an overview of our results. Besides the baseline, we show the results for only using the available training data (labeled *PoSTWITA*) and when adding the different types of external resources.

The baseline is not competitive to any of our system configurations, which confirms the generally poor performance of off-the-shelf PoS taggers on the social media domain. Using all resources yields our best result of 92.9%. Among the individual resources, word clusters perform best regarding overall accuracy as well as accuracy on out-of-vocabulary (OOV) tokens. This shows that clusters are also highly effective for Italian, as was previously shown for English (Owoputi et al., 2013) and German (Horsmann and Zesch, 2016).

We also computed the confidence interval by binomial normal approximation ($\alpha = 0.05$). We obtain an upper bound of 93.6 and a lower bound of 92.2. This shows that our best configuration is significantly better than using only the provided training data. Looking at the official PoSTWITA results, it also shows that there are no significant differences between the top-ranking systems.

Error Analysis In Table 2, we show the accuracy for each PoS tag on the test data set. The largest confusion class is between nouns and proper nouns, which is in line with previous findings for other languages (Horsmann and Zesch, 2016). It can be argued whether requiring the PoS tagger to make this kind of distinction is actually a good idea, as it often does not depend on syntactical properties, but on the wider usage context. Because of the high number of noun/proper confusions, it is also likely that improvements for this class will hide improvements on smaller classes that might be more important quality indicators for social media tagging. In our error analysis, we will thus focus on more interesting cases.

Tag	#	Acc	Primary Confusion
ADP_A	145	100.0	-
HASHTAG	115	100.0	-
MENTION	186	100.0	-
PUNCT	583	100.0	-
CONJ	123	99.2	VERB
URL	119	98.3	VERB
DET	306	95.8	PRON
ADP	351	95.7	ADV
PRON	327	93.3	DET
NUM	70	92.9	ADJ
INTJ	66	92.4	NOUN
NOUN	607	91.6	PROPN
VERB	568	91.6	AUX
AUX	109	90.8	VERB
ADV	321	90.3	SCONJ
SCONJ	60	90.0	PRON
ADJ	210	86.2	NOUN
EMO	79	83.5	SYM
PROPN	346	79.5	NOUN
VERB_CLIT	27	77.8	NOUN
SYM	12	72.7	PUNCT
X	27	55.6	EMO

Table 2: Accuracy per word class on the test data

In Table 3, we show examples of selected tagging errors. In case of the two adjective-determiner confusions both words occurred in the training data, but never as adjectives. The verb examples show cases where incorrectly tagging a verb as an auxiliary leads to a follow up error. We have to stress here that the feature set we use for training our PoS tagger does not use any linguistically knowledge about Italian. Thus, adding linguistically knowledge might help to better inform the tagger how to avoid such errors.

Amount of Training Data The amount of annotated social media text (120k tokens) in this

Adjective Confusions

Token	Gold/Pred	Token	Gold/Pred
cazzo	INTJ	successo	VERB
sono	VERB	dal	ADP_A
tutti	DET	quel	ADJ / DET
sti	ADJ / DET	cazzo	NOUN
tweet	NOUN	di	ADP

Verb Confusions

Token	Gold/Pred	Token	Gold/Pred
maggiormente	ADV	è	AUX / VERB
dell'	ADP_A	sempre	ADV
essere	VERB / AUX	stata	VERB / AUX
capito	ADJ / VERB	togliersi	VERB_CLIT
.	PUNCT	dai	ADP_A

Table 3: Adjective and Verb confusions

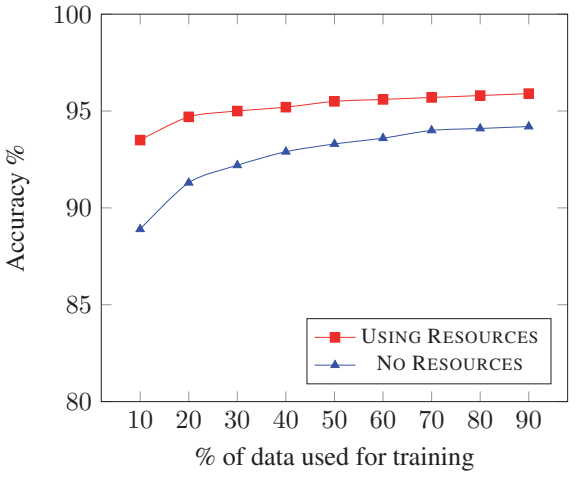


Figure 1: Learning Curve on training data with and without resources

shared-task is an order of magnitude larger than what was used in other shared tasks for tagging social media text. This raises the question of how much annotated training data is actually necessary to train a competitive social media PoS tagging model.

In Figure 1, we plot two learning curves that show how accuracy improves with an increasing amount of training data. We split the training data into ten chunks of equal size and add one additional data chunk in each iteration. We show two curves, one for just using the training data and one when additionally using all our resources. When using no resources, we see a rather steep and continuous increase of the learning curve which shows the challenges of the domain to provide sufficient training data. Using resources, this need of training data is compensated and only a small amount of training data is required to train a good model. The curves also show that the remaining problems are certainly not being solved by providing more training data.

4 Summary

We presented our contribution to the PoSTWITA shared task 2016 for PoS tagging of Italian social media text. We show that the same adaptation strategies that have been applied for English and German also lead to competitive results for Italian. Word clusters are the most effective resource and considerably help to reduce the problem of out-of-vocabulary tokens. In a learning curve experiment, we show that adding of more annotated data is not likely to provide further improvements and recommend instead to add more language specific knowledge. We make our experiments and resources publicly available.[2]

Acknowledgments

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant No. GRK 2167, Research Training Group "User-Centred Social Media".

References

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Peter F Brown, Peter V DeSouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18:467–479.

Richard Eckart de Castilho and Iryna Gurevych. 2014. A broad-coverage collection of portable NLP components for building shareable analysis pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT (OIAF4HLT) at COLING 2014*, pages 1–11, Dublin, Ireland.

Tobias Horsmann and Torsten Zesch. 2015. Effectiveness of Domain Adaptation Approaches for Social Media PoS Tagging. In *Proceeding of the 2nd Italian Conference on Computational Linguistics*, pages 166–170, Trento, Italy.

[2] https://github.com/Horsmann/EvalitaPoSTWITA2016.git

Tobias Horsmann and Torsten Zesch. 2016. LTL-UDE @ EmpiriST 2015: Tokenization and PoS Tagging of Social Media Text. In *Proceedings of the 10th Web as Corpus Workshop*, pages 120–126, Berlin, Germany.

John D Lafferty, Andrew McCallum, and Fernando C N Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA.

Olutobi Owoputi, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named Entity Recognition in Tweets: An Experimental Study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Stroudsburg, PA, USA.

Helmut Schmid. 1995. Improvements In Part-of-Speech Tagging With an Application To German. In *Proceedings of the ACL SIGDAT-Workshop*, pages 47–50.

Torsten Zesch and Tobias Horsmann. 2016. FlexTag: A Highly Flexible Pos Tagging Framework. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4259–4263, Portorož, Slovenia.

Mivoq Evalita 2016 PosTwITA tagger

Giulio Paci
Mivoq Srl
Padova - Italy
giulio.paci@mivoq.it

Abstract

English. The POS tagger developed by Mivoq to tag tweets according to PosTwITA task guidelines as defined at Evalita 2016 is presented. The system obtained third position with 92.7% of accuracy.

Italiano. *Si presenta il POS tagger sviluppato da Mivoq per etichettare i tweet secondo le linee guida del task PosTwITA, cos come definite per Evalita 2016. Il sistema ha ottenuto la terza posizione con un accuratezza del 92.7%.*

1 Introduction

Twitter messages (Tweets) are challenging for Natural Language Processing (NLP) due to the conversational nature of their content, the unconventional orthography and the 140 character limit of each tweet (Gimpel et al., 2011). Moreover tweets contain many elements that are not typical in conventional text, such as emoticons, hashtags, at-mentions, discourse markers, URL and emails.

Text-To-Speech (TTS) systems make large use of NLP technologies as part of input preprocessing, in order to cope with:

- homographs disambiguation: TTS systems may use POS tagging as a preliminary step to identify the correct pronunciation for those words that shares the same written form, but are pronounced differently. Many Italian homographs can be disambiguated using the POS tag (e.g., the string "ancora" has two possible pronunciations according to the fact that we are referring to the noun "anchor" or to the adverb "still"), although in some cases more information is needed;
- words expansion: as not all the text correspond to pronounceable words, TTS systems need to convert some strings into pronounceable words (e.g., numbers, units, acronyms, URL, ...). POS tags are useful to identify the function of a string and perform correct expansion (e.g., the string "1" can be expanded into "uno", "un" and "una", according to the POS tags of the surrounding strings);
- prosody prediction: prosody includes several aspects of speech, such as intonation, stress, tempo, rhythm and pauses, that are often not perfectly encoded by grammar or by choice of vocabulary, but still are important for the communication and should be correctly rendered by TTS systems. POS tags correlate with several prosodic aspects (e.g., content words are generally produced with more prominence than function words) and thus are useful for prosody prediction.

This work is the first attempt of the author to develop a POS tagger suitable for usage in a TTS system dealing with tweets.

2 Description of the system

The proposed system is the combination of several taggers and resources:

- Hunpos (Halácsy et al., 2007), an open-source reimplementation of TnT (Brants, 2000) HMM based tagger;
- Yamcha (Kudo and Matsumoto, 2003), an open-source Support Vector Machine (SVM) based tagger;
- CRFSuite (Okazaki, 2007), a Conditional Random Fields (CRF) based tagger;
- Evalita 2016 PosTwITA training data set;

- Evalita 2009 POS Tagging training data set (Attardi and Simi, 2009; Attardi et al., 2008; Zanchetta and Baroni, 2005), this corpus comprises 108,874 word forms divided into 3,719 sentences extracted from the online edition of the newspaper "La Repubblica" and annotated using the Tanl tag-set;

- ISTC pronunciation dictionary: originally developed for the Italian module of the Festival Text-To-Speech system (Cosi et al., 2001), has been later expanded by several contributors and currently includes pronunciations of 3,177,286 distinct word forms. POS tag information (using Tanl tag-set) has been added to each pronunciation for the purpose of pronunciation disambiguation; for this reason this information is reliable for all those words with multiple possible pronunciations, but many admissible tags may be missing for the other entries.

Six different taggers, corresponding to different combinations of these resources, have been tested in a 10-fold cross-validation scheme. Three taggers have been trained on the PosTwITA training data and thus can be used independently to solve the Evalita 2016 PosTwITA task. Two of them have been trained on Evalita 2009 Pos Tagging training data and can be used to solve that task instead. The sixth tagger combines the above taggers and is the system that has been proposed.

2.1 Hunpos

Hunpos has been used as a black box, without the use of an external morphological lexicon: an attempt have been made to use the ISTC pronunciation dictionary, but performance degraded. Hunpos has been trained on PosTwITA training data, where it obtained an average accuracy of 92.51%, and on Evalita 2009 Pos Tagging training data, where it obtained and average accuracy of 95.72%.

2.2 Yamcha

Yamcha allows the usage of arbitrary features and can be used to implement a wide range of taggers. Features combinations are implicitly expanded using a polynomial kernels and exploited by SVM (Kudo and Matsumoto, 2003).

Several feature sets have been tested, using the default parameters for Yamcha (i.e., only pair wise multi class method and second degree polynomial kernel have been used). Yamcha has been trained on PosTwITA training data and obtained an average accuracy of 95.41%.

2.2.1 Baseline

The baseline experiment with Yamcha consists in using features proposed by its author for English POS-tagging (Kudo, 2003 2005):

- the string to be annotated (i.e., the word);

- three Boolean flags set to true if: the string contains a digit, the string contains non alphanumeric characters, the first character of the string is an upper case character;

- the suffixes and prefixes of the string (with character length from 1 to 4, set to __nil__ if the original string is shorter than the suffix or the prefix length).

The default feature window has been used in this experiment (i.e., for each word form, features for previous two word forms and next two word forms are used, as long as the annotation results of the previous two word forms). Average accuracy is reported in table 1.

2.2.2 Twitter specific elements

A rule-based annotator for typical twitter elements (Prescott, 2012 2013) has been implemented:

- hashtags: an hash followed by a string composed by word characters only (the actual implementation allow some common symbols in the string, such as apostrophe, dots or &, thus matching author intention rather than proper twitter syntax);

- at-mentions: an optional dot, followed by an @ symbol, followed by a valid username (the actual implementation do not validate usernames and allows some common symbols in usernames);

- URLs (common mistakes are handled and matched in the implementation);

- emoticons: rules have been added to match both western (e.g., ":-)", ":-(", ...) and Asian (e.g., "^_^", "UwU", ...) style emoticons, to handle characters repetitions (e.g., ":-)))") and to match a subset of Unicode emoji. The rules have been tuned on a set of emoticons

described in Wikipedia (Wikipedia users, 2004 2016) and expanded according to the author's experience.

Although the accuracy improvement due to this feature was marginal (see table 1), it was present in all the tests and allowed almost perfect match of all Twitter specific elements, which is very important for words expansion.

2.2.3 Normalized string

Phonetic normalization has been proposed to deal with the many alternate spelling of words in English tweets (Gimpel et al., 2011). In this work a much simpler normalization is used, consisting in consecutive duplicated characters removal and converting to lower case. The following feature set has been tested:

- the string to be annotated (i.e., the word);
- three Boolean flags set to true if: the string contains a digit, the string contains non alphanumeric characters, the first character of the string is an upper case character;
- the suffixes and prefixes of the string (with character length from 1 to 3, set to __nil__ if the original string is shorter than the suffix or the prefix length);
- the prefixes and suffixes of the normalized string (with character length from 1 to 4 and 1 to 6 respectively).
- Twitter elements rule-based annotation.

In order to reduce the number of features, prefixes, suffixes and twitter annotations of the surrounding words has not been considered. The system achieved an average accuracy of 94.61%.

2.2.4 Dictionary tags

Finally 12 Boolean flags has been added, by performing a dictionary lookup using the normalized strings. Each flag corresponds to a PosTwITA tag (VERB_CLIT, VERB, INTJ, PROPN, NOUN, ADJ, ADP, ADP_A, SYM, ADV, DET, NUM) and is set to true if the ISTC dictionary contains a Tanl POS tag that can be mapped into it. By adding this feature the system achieved and average accuracy of 95.41%.

2.3 CRFSuite

The same feature sets used with Yamcha have been tested with CRFSuite, leading to very similar results, as shown in table 1. CRFSuite has been trained on both PosTwITA and on Evalita 2009 Pos Tagging training data sets, obtaining similar accuracy for both.

2.4 Tagger combination

The final system is a combination of five taggers based on Yamcha, by adding their output to the feature set. Tags associated to the surrounding tokens (3 previous and 3 next) are considered: using a larger window helped reducing errors with AUX and VERB tags. Results for individual taggers and the final system are shown in table 1. The system achieved an average accuracy of 95.97%. Implementing the same system using only the three taggers trained on PosTwITA data, lead to a very similar average accuracy of 95.74%, however the proposed system achieved better results in all the tests.

3 Results

	Hunpos	**Yamcha**	**CRFSuite**
	Evalita 2009 POS Tagging		
Hunpos	**95.72%**		
YN+T+D			**95.41%**
	Evalita 2016 PosTwITA		
Hunpos	**92.51%**		
YB		93.17%	93.02%
YB+T		93.30%	
YN+T		94.61%	94.17%
YN+T+D		**95.41%**	**95.31%**
MT		**95.97%**	

Table 1: 10-fold cross-validation average accuracy of a few configurations on both Evalita 2009 POS Tagging and Evalita 2016 PosTwITA training sets.

Table 1 reports average accuracy obtained in 10-fold cross-validation experiments on Evalita 2016 PosTwITA and Evalita 2009 POS Tagging data sets. Each column corresponds to a different tagger and each row corresponds to a different feature set, as described in section 2. YB is the baseline feature set described in section 2.2.1, YB+T is the baseline feature set with rule-based Twitter elements' annotation described in section 2.2.2, YN+T is the feature set described in section 2.2.3

	Hunpos	Yamcha	CRFSuite
Hunpos	**85.90%**		
	(86.43%)		
YB		88.95%	88.86%
		(89.75%)	(89.58%)
YB+T		88.91%	
		(89.72%)	
YN+T		90.10%	90.25%
		(91.01%)	(90.83%)
YN+T+D		**91.36%**	**92.06%**
		(92.27%)	(92.71%)
MT		**92.71%**	
		(93.74%)	

Table 2: blind test average accuracy of a few configurations.

Training set		*Test set*	
3362	.	124	,
2908	,	85	e
2315	**Monti**	82	.
2148	di	77	di
2109	:	66	che
1915	il	66	a
1652	e	64	”
1503	che	52	?
1499	a	50	:
1437	**governo**	49	...

Table 3: 10 most frequent tokens in PosTwITA training and test sets.

and YN+T+D is the YN+T feature set with the addition of dictionary usage as described in section 2.2.4. MT is the final system described in section 2.4. Table 2 reports accuracy results for the same configurations on the PosTwITA test set. In this case results after manual correction of the test set are reported below the official results.

4 Discussion

Results in table 1 and table 2 shows that Yamcha and CRFSuite behave very similarly. By using YN+T+D feature set, CRFSuite achieves accuracy similar to that of Hunpos on the Evalita 2009 POS Tagging training set. With that feature set, performance of CRFSuite on both Evalita 2009 POS Tagging and Evalita 2016 PosTwITA training sets are very similar, suggesting the idea that YN+T+D feature set is quite stable and can be used successfully for both tasks. It would be interesting to include similar features in Hunpos in order to confirm the hypothesis.

Results on the Evalita 2016 PosTwITA test set shows a big accuracy loss, suggesting a mismatch between the training and the test sets. Manual correction of the test set, performed by the author, alleviated the differences, but still results are not comparable. Table 3 reports the 10 most frequent tokens in the PosTwITA training and test sets. The test set includes only function words and punctuation, but the most frequent word in the training set is the proper noun “Monti” and the word “governo” (government) is also among the most frequent tokens. Including at-mentions, hashtags and without considering the case, the word “monti” appears in 3460 tokens, making it the most frequent token in the data set and suggesting a very narrow topic. On the other hand the test set topic seems more general: the most frequent tokens are either words or punctuation marks and the first proper noun, “Italia” (Italy), appears at position 43. Given the topic mismatch, the tagger combination seems more stable than individual taggers.

The author goal was to investigate the possibility to implement a POS tagger suitable for reading tweets within a TTS system. Confusing NOUN and PROPN tags, and confusing ADJ, NOUN, AUX and VERB tags (in particular with nouns derived from adjectives or with adjectives derived from verbs) are among the most frequent errors. These errors do not typically affect the pronunciations. Hashtags, at-mentions and URL are correctly recognized with just one error, so that correct expansion of these elements can be performed. Several emoticons were wrongly annotated as punctuation, due to the limited set of Unicode emoji recognized by the rule-based annotation system and can be easily fixed by extend the match to the whole Unicode emoji set.

The difference in terms of accuracy between CRFSuite with YN+T+D feature set and the tagger combination, does not seem to justify the overhead of running multiple taggers; it would be interesting to train the taggers on a more general data set, eventually using the proposed tagger to bootstrap its annotation. Assuming that the pronunciation dictionary is already available in the TTS, the YN+T+D feature set described in section 2 seems appropriate for the POS tagging task for both tweets and more conventional text.

References

Giuseppe Attardi and Maria Simi. 2009. Overview of the evalita 2009 part-of-speech tagging task. In *Poster and Workshop Proceedings of the 11th Conference of the Italian Association for Artificial Intelligence*, Reggio Emilia, Italy.

Giuseppe Attardi et al. 2008. Tanl (text analytics and natural language processing). URL: `http://medialab.di.unipi.it/wiki/SemaWiki`.

Thorsten Brants. 2000. Tnt: A statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, pages 224–231, Stroudsburg, PA, USA. Association for Computational Linguistics. doi:10.3115/974147.974178.

Piero Cosi, Fabio Tesser, Roberto Gretter, Cinzia Avesani, and Michael W. Macon. 2001. Festival speaks italian! In *7th European Conference on Speech Communication and Technology*.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 42–47, Stroudsburg, PA, USA. Association for Computational Linguistics. URL: `http://dl.acm.org/citation.cfm?id=2002736.2002747`.

Péter Halácsy, András Kornai, and Csaba Oravecz. 2007. Hunpos: An open source trigram tagger. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 209–212, Stroudsburg, PA, USA. Association for Computational Linguistics. URL: `http://dl.acm.org/citation.cfm?id=1557769.1557830`.

Taku Kudo and Yuji Matsumoto. 2003. Fast methods for kernel-based text analysis. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 24–31, Stroudsburg, PA, USA. Association for Computational Linguistics. doi:10.3115/1075096.1075100.

Taku Kudo. 2003-2005. Yamcha: Yet another multipurpose chunk annotator. URL: `http://chasen.org/~taku/software/yamcha/`.

Naoaki Okazaki. 2007. Crfsuite: a fast implementation of conditional random fields (crfs). URL: `http://www.chokkan.org/software/crfsuite/`.

Adam Prescott. 2012-2013. twitter-format - syntax and conventions used in twitter statuses. URL: `http://aprescott.github.io/twitter-format/twitter-format.7`.

Wikipedia users. 2004-2016. Emoticon. In *Wikipedia*. URL: `https://it.wikipedia.org/wiki/Emoticon`.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! a free corpus-based morphological resource for the italian language. In *PROCEEDINGS OF CORPUS LINGUISTICS*. URL: `http://dev.sslmit.unibo.it/linguistics/morph-it.php`.

NLP–NITMZ:Part–of–Speech Tagging on Italian Social Media Text using Hidden Markov Model

Partha Pakray
Deptt. of Computer Science & Engg.
National Institute of Technology
Mizoram, Aizawl, India
parthapakray@gmail.com

Goutam Majumder
Deptt. of Computer Science & Engg.
National Institute of Technology
Mizoram, Aizawl, India
goutam.nita@gmail.com

Abstract

English. This paper describes our approach on Part-of-Speech tagging for Italian Social Media Texts (PoSTWITA), which is one of the task of EVALITA 2016 campaign. EVALITA is a evaluation campaign, where teams are participated and submit their systems towards the developing of tools related to Natural Language Processing (NLP) and Speech for Italian language. Our team **NLP–NITMZ** participated in the PoS tagging challenge for Italian Social Media Texts. In this task, total 9 team was participated and out of 4759 tags **Team1** successfully identified 4435 tags and get the 1^{st} rank. Our team get the 8^{th} rank officially and we successfully identified 4091 tags as a accuracy of 85.96%.

Italiano. *In questo articolo descriviamo la nostra partecipazione al task di tagging for Italian Social Media Texts (PoSTWITA), che uno dei task della campagna Evalita 2016. A questo task hanno partecipato 9 team; su 4759 tag il team vincitore ha identificato correttamante 4435 PoS tag. Il nostro team si classificato all'ottavo posto con 4091 PoS tag annotati correttamente ed una percentuale di accuratezza di 85.96*

1 Introduction

EVALITA is a evaluation campaign, where researchers are contributes tools for Natural Language Processing (NLP) and Speech for Italian language. The main objective is to promote the development of language and speech technologies by shared framework, where different systems and approaches can be evaluated. EVALITA 2016, is the 5^{th} evaluation campaign, where following six tasks are organized such as:

- ArtiPhon – Articulatory Phone Recognition
- FactA – Event Factuality Annotation
- NEEL–IT – Named Entity Recognition and Linking in Italian Tweets
- PoSTWITA – POS tagging for Italian Social Media Texts
- QA4FAQ – Question Answering for Frequently Asked Questions
- SENTIPOLC – SENTIment POLarity Classification

In addition, a new challenge to this event is also organized by IBM Italy as *IBM Watson Services Challenge*. Among these challenges our team NLP–NITMZ is participated in 4^{th} task i.e. POS tagging for Italian Social Media Texts (PoSTWITA).

The main concern about PosTWITA is, Part-of-Speech (PoS) tagging for automatic evaluation of social media texts, in particular for micro–blogging texts such as tweets, which have many application such as identifying trends and upcoming events in various fields. For these applications NLP based methods need to be adapted for obtaining a reliable processing of text. In literature various attempts were already taken for developing of such specialised tools (Derczynski et al., 2013), (Neunerdt et al., 2013), (Pakray et al., 2015), (Majumder et al., 2016) for other languages, but for Italian is lack of such resources both regarding annotated corpora and specific PoS–tagging tools. For these reasons, EVALITA 2016 proposes the domain adaptation of PoS–taggers to Twitter texts.

For this task, we used a supervised leaning for PoS tagging and the details of system implementation is given in section 2. We discuss the per-

formance of the system in section 3. Finally, we conclude our task in section 4.

2 Proposed Method

For this task, we used supervised learning approach to build the model. First we implement the conditional model for PoS tagging and then to simplify the model we used Bayesian classification based generative model. Further this generative model is simplified based on two key assumptions to implement the HMM model using bigram.

2.1 Conditional Model Approach

In machine learning supervised problems are defined as a set of input called training examples $(x^{(1)}, y^{(1)}) \cdots (x^{(m)}, y^{(m)})$, where each input $x^{(i)}$ paired with a output label $y^{(i)}$. In this task, our goal is to learn a function $f : X \rightarrow Y$, where X and Y refers to the set of possible input and labels.

For PoS tagging problem, each input represents a sequence of words $x_1^{(i)}, \cdots, x_{n_i}^{(i)}$ and labels be a sequence of tags $y_1^{(i)}, \cdots, y_{n_i}^{(i)}$, where n_i refers to the length of i^{th} training example. In this machine learning each input x be a sentence of Italian language and each label be the possible PoS tag. We use conditional model to define the function $f(x)$ and we define the conditional probability as

$$p\left(y|x\right)$$

for any x, y pair. We use training examples to estimate the parameters of the model and output of the model for a given test example x is measured as

$$f(x) = arg \max_{y \in Y} p(y|x) \tag{1}$$

Thus we consider the most likely label y as the output of the trained model. If the model $p(y|x)$ is close to the true conditional distribution of a labels given inputs, so the function $f(x)$ will consider as an optimal.

2.2 Generative Model

In this model, we use the Bayes' rule to transform the Eq.1 into a set of other probabilities called *generative model*. Without estimating the conditional probability $p(y|x)$, in generative model we use the Bayesian classification

$$p(x, y)$$

over (x, y) pairs. In this case, we further break down the probability $p(x, y)$ as follows:

$$p(x, y) = p(y)p(x|y) \tag{2}$$

and then we estimate the model $p(y)$ and $p(x|y)$ separately. We consider $p(y)$ as a *prior* probability distribution over label y and $p(x|y)$ is the probability of generating the input x, given that the underlying label is y.

We use the Bayes rule to derive the conditional probability $p(y|x)$ for any (x, y) pair:

$$p(y|x) = \frac{p(y)p(x|y)}{p(x)} \tag{3}$$

where

$$p(x) = \sum_{y \in Y} p(x, y) = \sum_{y \in Y} p(y)p(x|y) \tag{4}$$

We apply Bayes rule directly to a new test example x, so the output of the model $f(x)$, can be estimated as follows:

$$f(x) = arg \max_{y} p(y)p(x|y) \tag{5}$$

To simplify Eq.5, we use Hidden Markov Model (HMM) taggers with two simplifying assumptions. The first assumption is that the probability of word appearing depends only on its own PoS tag as follows:

$$p(w_1^n \; t_1^n) \approx \prod_{i=1}^{n} p(w_i \; t_i) \tag{6}$$

where $p(w_1^n \; t_1^n)$ means probability of tag t_i with word w_i. The second assumption is that the probability of a tag appearing is dependent only on the previous tag, rather than entire tag sequence. This is known as bigram assumption and can be measured as follows:

$$p(t_1^n) \approx \prod_{i=1}^{n} p(t_i, t_{i-1}) \tag{7}$$

Further, we incorporate these two assumptions in Eq.5 by which a bigram tagger estimates the most probable tag as follows:

$$\hat{t_1^n} = arg \max_{t_1^n} p(t_1^n \; w_1^n) \approx$$
$$arg \max_{t_1^n} \prod_{i=1}^{n} p(w_i \; t_i)p(t_i \; t_{i-1}) \tag{8}$$

3 Experiment Results

3.1 Dataset

For the proposed task organizers re-uses the tweets being part of the EVALITA2014 SENTIPLOC corpus. Both the development and test set first annotated manually for a global amount of $4,041$ and $1,749$ tweets and distributed as the new development set. Then a new manually annotated test set, which is composed of 600 and 700 tweets were produced using texts from the same period of time. All the annotations are carried out by three different annotators. Further a tokenised version of the texts is also distributed in order to avoid tokenisation problems among participants and the boring problem of disappeared tweets.

3.2 Results

For this task, total 13 runs were submitted 9 teams and among these runs 4 Unofficial runs also submitted. In Table 1 we list out all results for this task.

Rank	Team	Successful Tags	Accuracy
1	Team1	4435	93.19
2	Team2	4419	92.86
3	Team3	4416	92.79
4	Team4	4412	92.70
5	Team3	4400	92.46
6	Team5	4390	92.25
7	Team5	4371	91.85
8	Team6	4358	91.57
9	Team6	4356	91.53
10	Team7	4183	87.89
11	**Team8**	**4091**	**85.96**
12	Team2	3892	81.78
13	Team9	3617	76.00

Table 1: Tagging Accuracy of Participated Teams

Team 2, 3, 5 and 6 submitted one Un-Official run with compulsory one and these Un-Official submissions are ranked as 12^{th}, 3^{rd}, 7^{th} and 9^{th} respectively. We also listed these submissions in Table 1 with other runs. Our team **NLP–NITMZ** represent as **Team8** and ranked as 11^{th} in this task.

3.3 Comparison with other submissions

In this competition, a total of 4759 words were given for tagging purpose. These words were categories into 22 PoS tags and our team successfully tags 4091 words with 668 unsuccessful tags. The 1^{st} ranked team successfully tags 4435 words and the last positioned team i.e. Team9 successfully identified 3617 tags. In Table 2, we provide our system tag wise statistics.

Sl. No.	Tag	Successful Tags
1	PRON	292
2	AUX	82
3	PROPN	283
4	EMO	30
5	SYM	8
6	NUM	63
7	ADJ	145
8	SCONJ	37
9	ADP	332
10	URL	117
11	DET	288
12	HASHTAG	114
13	ADV	281
14	VERB_CLIT	10
15	PUNCT	582
16	VERB	443
17	CONJ	122
18	X	3
19	INTJ	50
20	MENTION	186
21	ADP_A	144
22	NOUN	479

Table 2: Tag wise Statistics of NLP–NITMZ Team

4 Conclusion

This PoS tagging task of EVALITA 2016 campaign is for Italian language and our system ranked 11^{th} position for the task of POS tagging for Italian Social Media Texts. We also want to mentioned that, authors are not native speaker of the Italian language. We build a supervised learning model based on the available knowledge on training dataset.

Acknowledgements

This work presented here under the research project Grant No. YSS/2015/000988 and supported by the Department of Science & Technology (DST) and Science and Engineering Research Board (SERB), Govt. of India. Authors are also acknowledges the Department of Computer Science & Engineering of National Institute of Tech-

nology Mizoram, India for proving infrastructural facilities.

References

Derczynski, Leon, Alan Ritter, Sam Clark, and Kalina Bontcheva. 2013. Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data. In *RANLP*, pages 198–206.

Neunerdt Melanie, Bianka Trevisan, Michael Reyer, and Rudolf Mathar. 2013. Part-of-speech tagging for social media texts. In *Language Processing and Knowledge in the Web*, pages 139–150, Springer Berlin Heidelberg.

Partha Pakray, Arunagshu Pal, Goutam Majumder, and Alexander Gelbukh. 2015. Resource Building and Parts-of-Speech (POS) Tagging for the Mizo Language. In *Fourteenth Mexican International Conference on Artificial Intelligence (MICAI)*, pages 3–7. IEEE, October.

Goutam Majumder, Partha Pakray and Alexander Gelbukh. 2016. Literature Survey: Multiword Expressions (MWE) for Mizo Language. In *17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, to be published as an issue of Lecture Notes in Computer Science, Springer. Konya, Turkey. April.

When silver glitters more than gold: Bootstrapping an Italian part-of-speech tagger for Twitter

Barbara Plank
University of Groningen
The Netherlands
b.plank@rug.nl

Malvina Nissim
University of Groningen
The Netherlands
m.nissim@rug.nl

Abstract

English. We bootstrap a state-of-the-art part-of-speech tagger to tag Italian Twitter data, in the context of the Evalita 2016 PoSTWITA shared task. We show that training the tagger on native Twitter data enriched with little amounts of specifically selected gold data and additional silver-labelled data scraped from Facebook, yields better results than using large amounts of manually annotated data from a mix of genres.

Italiano. *Nell'ambito della campagna di valutazione PoSTWITA di Evalita 2016, addestriamo due modelli che differiscono nel grado di supervisione in fase di training. Il modello addestrato con due cicli di bootstrapping usando post da Facebook, e che quindi impara anche da etichette "silver", ha una performance superiore alla versione supervisionata che usa solo dati annotati manualmente. Discutiamo l'importanza della scelta dei dati di training e development.*

1 Introduction

The emergence and abundance of social media texts has prompted the urge to develop tools that are able to process language which is often non-conventional, both in terms of lexicon as well as grammar. Indeed, models trained on standard newswire data heavily suffer when used on data from a different language variety, especially Twitter (McClosky et al., 2010; Foster et al., 2011; Gimpel et al., 2011; Plank, 2016).

As a way to equip microblog processing with efficient tools, two ways of developing Twitter-compliant models have been explored. One option is to transform Twitter language back to what pre-trained models already know via normalisation operations, so that existing tools are more successful on such different data. The other option is to create *native* models by training them on labelled Twitter data. The drawback of the first option is that it's not clear what norm to target: "what is standard language?" (Eisenstein, 2013; Plank, 2016), and implementing normalisation procedures requires quite a lot of manual intervention and subjective decisions. The drawback of the second option is that manually annotated Twitter data isn't readily available, and it is costly to produce.

In this paper, we report on our participation in PoSTWITA[1], the EVALITA 2016 shared task on Italian Part-of-Speech (POS) tagging for Twitter (Tamburini et al., 2016). We emphasise an approach geared to building a *single model* (rather than an ensemble) based on weakly supervised learning, thus favouring (over normalisation) the aforementioned second option of learning *invariant representations*, also for theoretical reasons. We address the bottleneck of acquiring manually annotated data by suggesting and showing that a semi-supervised approach that mainly focuses on tweaking data selection within a bootstrapping setting can be successfully pursued for this task. Contextually, we show that large amounts of manually annotated data might not be helpful if data isn't "of the right kind".

2 Data selection and bootstrapping

In adapting a POS tagger to Twitter, we mainly focus on ways of selectively enriching the training set with additional data. Rather than simply adding large amounts of existing annotated data, we investigate ways of selecting smaller amounts of more appropriate training instances, possibly even tagged with silver rather than gold labels. As

[1] http://corpora.ficlit.unibo.it/PoSTWITA/

for the model itself, we simply take an off-the-shelf tagger, namely a bi-directional Long Short-Term Memory (bi-LSTM) model (Plank et al., 2016), which we use with default parameters (see Section 3.2) apart from initializing it with Twitter-trained embeddings (Section 3.1).

Our first model is trained on the PoSTWITA training set plus additional gold data selected according to two criteria (see below: *Two shades of gold*). This model is used to tag a collection of Facebook posts in a bootstrapping setting with two cycles (see below: *Bootstrapping via Facebook*). The rationale behind using Facebook as *not-so-distant* source when targeting Twitter is the following: many Facebook posts of public, non-personal pages resemble tweets in style, because of brevity and the use of hashtags. However, differently from random tweets, they are usually correctly formed grammatically and spelling-wise, and often provide more context, which allows for more accurate tagging.

Two shades of gold We used the Italian portion of the latest release (v1.3) of the Universal Dependency (UD) dataset (Nivre et al., 2016), from which we extracted two subsets, according to two different criteria. First, we selected data on the basis of its *origin*, trying to match the Twitter training data as close as possible. For this reason, we used the Facebook subportion (`UD_FB`). These are 45 sentences that presumably stem from the Italian Facebook help pages and contain questions and short answers.[2] Second, by looking at the confusion matrix of one of the initial models, we saw that the model's performance was especially poor for cliticised verbs and interjections, tags that are also infrequent in the training set (Table 2). Therefore, from the Italian UD portion we selected *any* data (in terms of origin/genre) which contained the `VERB_CLIT` or `INTJ` tag, with the aim to boost the identification of these categories. We refer to this set of 933 sentences as `UD_verb_clit+intj`.

Bootstrapping via Facebook We augmented our training set with silver-labelled data. With our best model trained on the original task data plus `UD_verb_clit+intj` and `UD_FB`, we tagged a collection of Facebook posts, added those to the training pool, and retrained our tagger. We used two iterations of indelible self-training (Abney, 2007), i.e., adding automatically tagged data where labels do not change once added. Using the Facebook API through the Facebook-sdk python library[3], we scraped an average of 100 posts for each of the following pages, selected on the basis of our intuition and on reasonable site popularity:

- sport: `corrieredellosport`
- news: `Ansa.it`, `ilsole24ore`, `lastampa.it`
- politics: `matteorenziufficiale`
- entertainment: `novella2000`, `alFemminile`
- travel: `viaggiart`

We included a second cycle of bootstrapping, scraping a few more Facebook pages (`soloGossip.it`, `paesionline`, `espressonline`, `LaGazzettaDelloSport`, again with an average of 100 posts each), and tagging the posts with the model that had been re-trained on the original training set plus the first round of Facebook data with silver labels (we refer to the whole of the automatically-labelled Facebook data as `FB_silver`). `FB_silver` was added to the training pool to train the final model. Statistics on the obtained data are given in Table 1.[4]

Table 1: Statistics on the additional datasets.

Data	Type	Sents	Tokens
`UD_FB`	gold	45	580
`UD_verb_clit+intj`	gold	933	26k
`FB` (all, iter 1)	silver	2243	37k
`FB` (all, iter 2)	silver	3071	47k
Total added data	gold+silver	4049	74k

3 Experiments and Results

In this section we describe how we developed the two models of the final submission, including all preprocessing decisions. We highlight the importance of choosing an adequate development set to identify promising directions.

3.1 Experimental Setup

PoSTWITA data In the context of PoSTWITA, training data was provided to all participants in the

[2] These are labelled as `4-FB` in the comment section of UD. Examples include: *Prima di effettuare la registrazione. È vero che Facebook sarà a pagamento?*

[3] https://pypi.python.org/pypi/facebook-sdk

[4] Due to time constraints we did not add further iterations; we cannot judge if we already reached a performance plateau.

Table 2: Tag distribution in the original trainset.

Tag	Explanation	#Tokens	Example
NOUN	noun	16378	cittadini
PUNCT	punctuation	14513	?
VERB	verb	12380	apprezzo
PROPN	proper noun	11092	Ancona
DET	determiner	8955	il
ADP	preposition	8145	per
ADV	adverb	6041	sempre
PRON	pronoun	5656	quello
ADJ	adjective	5494	mondiale
HASHTAG	hashtag	5395	#manovra
ADP_A	articulated prep	4465	nella
CONJ	coordinating conj	2876	ma
MENTION	mention	2592	@InArteMorgan
AUX	auxiliary verb	2273	potrebbe
URL	url	2141	http://t.co/La3opKcp
SCONJ	subordinating conj	1521	quando
INTJ	interjection	1404	fanculo
NUM	number	1357	23%
X	anything else	776	s...
EMO	emoticon	637	😳
VERB_CLIT	verb+clitic	539	vergognarsi
SYM	symbol	334	→
PART	particle	3	's

form of manually labelled tweets. The tags comply with the UD tagset, with a couple of modifications due to the specific genre (emoticons are labelled with a dedicated tag, for example), and subjective choices in the treatment of some morphological traits typical of Italian. Specifically, clitics and articulated prepositions are treated as one single form (see below: *UD fused forms*). The training set contains 6438 tweets, for a total of ca. 115K tokens. The distribution of tags together with examples is given in Table 2. The test set comprises 301 tweets (ca. 4800 tokens).

UD fused forms In the UD scheme for Italian, articulated prepositions (ADP_A) and cliticised verbs (VERB_CLIT) are annotated as separate word forms, while in PoSTWITA the original word form (e.g., 'alla' or 'arricchirsi') is annotated as a whole. In order to get the PoSTWITA ADP_A and VERB_CLIT tags for these fused word forms from UD, we adjust the UCPH ud-conversion-tools[5] (Agić et al., 2016) that propagates head POS information up to the original form.

Pre-processing of unlabelled data For the Facebook data, we use a simplistic off-the-shelf rule-based tokeniser that segments sentences by punctuation and tokens by whitespace.[6] We normalise URLs to a single token (http://www.someurl.org) and add a rule for smileys. Finally, we remove sentences from the Facebook data were more than 90% of the tokens are in all caps. Unlabelled data used for embeddings is preprocessed only with normalisation of usernames and URLs.

Word Embeddings We induced word embeddings from 5 million Italian tweets (TWITA) from Twita (Basile and Nissim, 2013). Vectors were created using word2vec (Mikolov and Dean, 2013) with default parameters, except for the fact that we set the dimensions to 64, to match the vector size of the multilingual (POLY) embeddings (Al-Rfou et al., 2013) used by Plank et al. (2016). We dealt with unknown words by adding a "UNK" token computing the mean vector of three infrequent words ("vip!","cuora", "White").

Figure 1: Word cloud from the training data.

Creation of a *realistic* internal development set The original task data is distributed as a single training file. In initial experiments we saw that performance varied considerably for different random subsets. This was due to a large bias towards tweets about 'Monti' and 'Grillo', see Figure 1, but also because of duplicate tweets. We opted to create *the most difficult* development set possible. This development set was achieved by removing duplicates, and randomly selecting a subset of tweets that do not mention 'Grillo' or 'Monti' while maximizing out-of-vocabulary (OOV) rate with respect to the training data. Hence, our internal development set consisted of 700 tweets with an OOV approaching 50%. This represents a more realistic testing scenario. Indeed, the baseline (the basic bi-LSTM model), dropped from 94.37 to 92.41 computed on the earlier development set were we had randomly selected 1/5 of the data, with an OOV of 45% (see Table 4).

3.2 Model

The bidirectional Long Short-Term Memory model bilty[7] is illustrated in Figure 2. It is a

[5]https://github.com/coastalcph/ud-conversion-tools

[6]https://github.com/bplank/multilingualtokenizer

[7]https://github.com/bplank/bilstm-aux

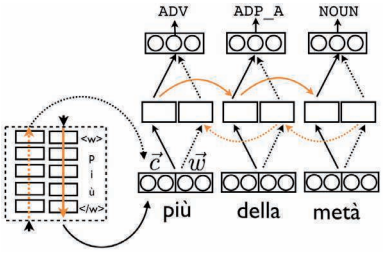


Figure 2: Hierarchical bi-LSTM model using word $\vec{w}$ and character $\vec{c}$ representations.

context bi-LSTM taking as input word embeddings $\vec{w}$. Character embeddings $\vec{c}$ are incorporated via a hierarchical bi-LSTM using a sequence bi-LSTM at the lower level (Ballesteros et al., 2015; Plank et al., 2016). The character representation is concatenated with the (learned) word embeddings $\vec{w}$ to form the input to the context bi-LSTM at the upper layers. We took default parameters, i.e., character embeddings set to 100, word embeddings set to 64, 20 iterations of training using Stochastic Gradient Descent, a single bi-LSTM layer and regularization using Gaussian noise with $\sigma = 0.2$ (`cdim 100, trainer sgd, indim 64, iters 20, h_layer 1, sigma 0.2`). The model has been shown to achieve state-of-the-art performance on a range of languages, where the incorporation of character information was particularly effective (Plank et al., 2016). With these features and settings we train two models on different training sets.

GOLDPICK `bilty` with pre-initialised TWITA embeddings, trained on the PoSTWITA training set plus selected gold data (`UD_FB` + `UD_verb_clit+intj`).

SILVERBOOT a bootstrapped version of GOLDPICK, where `FB_silver` (see Section 2) is also added to the training pool, which thus includes both gold and silver data.

3.3 Results on test data

Participants were allowed to submit one official, and one additional (unofficial) run. Because on development data SILVERBOOT performed better than GOLDPICK, we selected the former for our official submission and the latter for the unofficial one, making it thus also possible to assess the specific contribution of bootstrapping to performance.

Table 3: Results on the official test set. BEST is the highest performing system at PoSTWITA.

System	Accuracy
BEST	93.19
SILVERBOOT (official)	92.25
GOLDPICK (unofficial)	91.85
TNT (on POSTWITA train)	84.83
TNT (on SILVERBOOT data)	85.52

Table 3 shows the results on the official test data for both our models and TNT (Brants, 2000). The results show that adding bootstrapped silver data outperforms the model trained on gold data alone. The additional training data included in SILVERBOOT reduced the OOV rate for the test-set to 41.2% (compared to 46.9% with respect to the original PoSTWITA training set). Note that, on the original randomly selected development set the results were less indicative of the contribution of the silver data (see Table 4), showing the importance of a carefully selected development set.

4 What didn't work

In addition to what we found to boost the tagger's performance, we also observed what didn't yield any improvements, and in some case even lowered global accuracy. What we experimented with was triggered by intuition and previous work, as well as what we had already found to be successful, such as selecting additional data to make up for under-represented tags in the training set. However, everything we report in this section turned out to be either pointless or detrimental.

More data We added to the training data *all* (train, development, and test) sections from the Italian part of UD1.3. While training on selected gold data (978 sentences) yielded 95.06% accuracy, adding all of the UD-data (12k sentences of newswire, legal and wiki texts) yielded a disappointing 94.88% in initial experiments (see Table 4), also considerably slowing down training.

Next, we tried to add more Twitter data from XLIME, a publicly available corpus with multiple layers of manually assigned labels, including POS tags, for a total of ca. 8600 tweets and 160K tokens (Rei et al., 2016). The data isn't provided as a single gold standard file but in the form of

Table 4: Results on internal development set.

System	Accuracy
Internal dev (prior) OOV: 45%	
BASELINE (w/o emb)	94.37
+POLY emb	94.15
+TWITA emb	**94.69**
BASELINE+TWITA emb	
+Morphit! coarse MTL	94.61
+Morphit! fine MTL	94.68
+UD all	94.88
+gold-picked	95.06
+gold-picked+silver (1st round)	**95.08**
Internal dev (realistic) OOV: 50%	
BASELINE (incl. TWITA emb)	92.41
+gold (GOLDPICK)	93.19
+gold+silver (SILVERBOOT)	**93.42**
adding more gold (Twitter) data:	
+XLIME ADJUDICATED (48)	92.58
+XLIME SINGLE ANNOT.	91.67
+XLIME ALL (8k)	92.04

separate annotations produced by different judges, so that we used MACE (Hovy et al., 2013) to adjudicate divergences. Additionally, the tagset is slightly different from the UD set, so that we had to implement a mapping. The results in Table 4 show that adding all of the XLIME data declines performance, despite careful preprocessing to map the tags and resolve annotation divergences.

More tag-specific data From the matrix computed on the dev set, it emerged that the most confused categories were `NOUN` and `PROPN`. Following the same principle that led us to add `UD_verb_clit+intj`, we tried to reduce such confusion by providing additional training data containing proper nouns. This did not yield any improvements, neither in terms of global accuracy, nor in terms of precision and recall of the two tags.

Multi-task learning Multi-task learning (MTL) (Caruana, 1997), namely a learning setting where more than one task is learnt at the same time, has been shown to improve performance for several NLP tasks (Collobert et al., 2011; Bordes et al., 2012; Liu et al., 2015). Often, what is learnt is one main task and, additionally, a number of auxiliary tasks, where the latter should help the model converge better and overfit less on the former. In this context, the additional signal we use to support the learning of each token's POS tag is the token's degree of ambiguity. Using the information stored in *Morph-it!*, a lexicon of Italian inflected forms with their lemma and morphological features (Zanchetta and Baroni, 2005), we obtained the *number* of all different tags potentially associated to each token. Because the *Morph-it!* labels are highly fine-grained we derived two different ambiguity scores, one on the original and one on coarser tags. In neither case the additional signal contributed to the tagger's performance, but we have not explored this direction fully and leave it for future investigations.

5 Conclusions

The main conclusion we draw from the experiments in this paper is that *data selection matters*, not only for training but also while developing for taking informed decisions. Indeed, only after creating a carefully designed internal development set we obtained stronger evidence of the contribution of silver data which is also reflected in the official results. We also observe that choosing less but more targeted data is more effective. For instance, TWITA embeddings contribute more than generic POLY embeddings which were trained on substantially larger amounts of Wikipedia data. Also, just blindly adding training data does not help. We have seen that using the whole of the UD corpus is not beneficial to performance when compared to a small amount of selected gold data, both in terms of origin and labels covered. Finally, and most importantly, we have found that adding little amounts of not-so-distant silver data obtained via bootstrapping resulted in our best model.

We believe the low performance observed when adding xLIME data is likely due to the non-correspondence of tags in the two datasets, which required a heuristic-based mapping. While this is only a speculation that requires further investigation, it seems to indicate that exploring semi-supervised strategies is preferrable to producing idiosyncratic or project-specific gold annotations.

Acknowledgments We thank the CIT of the University of Groningen for providing access to the Peregrine HPC cluster. Barbara Plank acknowledges NVIDIA corporation for support.

References

Steven Abney. 2007. *Semisupervised learning for computational linguistics*. CRC Press.

Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. Multilingual projection for parsing truly low-resource languages. *Transactions of the Association for Computational Linguistics (TACL)*, 4:301–312.

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual NLP. *arXiv preprint arXiv:1307.1662*.

Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with lstms. In *EMNLP*.

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2012. Joint learning of words and meaning representations for open-text semantic parsing. In *AISTATS*, volume 351, pages 423–424.

Thorsten Brants. 2000. Tnt: a statistical part-of-speech tagger. In *ANLP*.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28:41–75.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Jacob Eisenstein. 2013. What to do about bad language on the internet. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 359–369, Atlanta.

Jennifer Foster, Ozlem Cetinoglu, Joachim Wagner, Josef Le Roux, Joakim Nivre, Deirde Hogan, and Josef van Genabith. 2011. From news to comments: Resources and benchmarks for parsing the language of Web 2.0. In *IJCNLP*.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *Proceedings of ACL*.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *NAACL*.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proc. NAACL*.

David McClosky, Eugene Charniak, and Mark Johnson. 2010. Automatic domain adaptation for parsing. In *NAACL-HLT*.

T Mikolov and J Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*.

Joakim Nivre et al. 2016. Universal dependencies 1.3. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University in Prague.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *ACL*.

Barbara Plank. 2016. What to do about non-standard (or non-canonical) language in NLP. In *KONVENS*.

Luis Rei, Dunja Mladenic, and Simon Krek. 2016. A multilingual social media linguistic corpus. In *Conference of CMC and Social Media Corpora for the Humanities*.

Fabio Tamburini, Cristina Bosco, Alessandro Mazzei, and Andrea Bolioli. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! A free corpus-based morphological resource for the Italian language. *Corpus Linguistics 2005*, 1(1).

bot.zen @ EVALITA 2016 - A minimally-deep learning PoS-tagger (trained for Italian Tweets)

Egon W. Stemle
Institute for Specialised Communication and Multilingualism
EURAC Research
Bolzano/Bozen, Italy
egon.stemle@eurac.edu

Abstract

English. This article describes the system that participated in the *POS tagging for Italian Social Media Texts* (PoSTWITA) task of the 5th periodic evaluation campaign of Natural Language Processing (NLP) and speech tools for the Italian language EVALITA 2016.

The work is a continuation of Stemle (2016) with minor modifications to the system and different data sets. It combines a small assertion of trending techniques, which implement matured methods, from NLP and ML to achieve competitive results on PoS tagging of Italian Twitter texts; in particular, the system uses word embeddings and character-level representations of word beginnings and endings in a LSTM RNN architecture. Labelled data (Italian UD corpus, DiDi and PoSTWITA) and unlabbelled data (Italian C4Corpus and PAISÀ) were used for training.

The system is available under the APLv2 open-source license.

Italiano. *Questo articolo descrive il sistema che ha partecipato al task* POS tagging for Italian Social Media Texts (PoSTWita) *nell'ambito di EVALITA 2016, la 5° campagna di valutazione periodica del Natural Language Processing (NLP) e delle tecnologie del linguaggio.*

Il lavoro è un proseguimento di quanto descritto in Stemle (2016), con modifiche minime al sistema e insiemi di dati differenti. Il lavoro combina alcune tecniche correnti che implementano metodi comprovati dell'NLP e del Machine Learning, per raggiungere risultati competitivi nel PoS tagging dei testi italiani di Twitter. In particolare il sistema utilizza strategie di word embedding e di rappresentazione character-level di inizio e fine parola, in un'architettura LSTM RNN. Dati etichettati (Italian UD corpus, DiDi e PoSTWITA) e dati non etichettati (Italian C4Corpus e PAISÀ) sono stati utilizzati in fase di training.

Il sistema è disponibile sotto licenza open source APLv2.

1 Introduction

Part-of-speech (PoS) tagging is an essential processing stage for virtually all NLP applications. Subsequent tasks, like parsing, named-entity recognition, event detection, and machine translation, often utilise PoS tags, and benefit (directly or indirectly) from accurate tag sequences.

Actual work on PoS tagging, meanwhile, mainly concentrated on standardized texts for many years, and frequent phenomena in computer-mediated communication (CMC) and Web corpora such as emoticons, acronyms, interaction words, iteration of letters, graphostylistics, shortenings, addressing terms, spelling variations, and boilerplate (Androutsopoulos, 2007; Bernardini et al., 2008; Beißwenger, 2013) still deteriorate the performance of PoS-taggers (Giesbrecht and Evert, 2009; Baldwin et al., 2013).

On the other hand, the interest in automatic evaluation of social media texts, in particular for microblogging texts such as tweets, has been growing considerably, and specialised tools for

Twitter data have become available for different languages. But Italian completely lacks such resources, both regarding annotated corpora and specific PoS-tagging tools.[1] To this end, the *POS tagging for Italian Social Media Texts* (PoSTWITA) task was proposed for EVALITA 2016 concerning the domain adaptation of PoS-taggers to Twitter texts.

Our system combined `word2vec` (`w2v`) word embeddings (WEs) with a single-layer Long Short Term Memory (LSTM) recurrent neural network (RNN) architecture. The sequence of unlabelled `w2v` representations of words is accompanied by the sequence of n-grams of the word beginnings and endings, and is fed into the RNN which in turn predicts PoS labels.

The paper is organised as follows: We present our system design in Section 2, the implementation in Section 3, and its evaluation in Section 4. Section 5 concludes with an outlook on possible implementation improvements.

2 Design

Overall, our design takes inspiration from as far back as Benello et al. (1989) who used four preceding words and one following word in a feed-forward neural network with backpropagation for PoS tagging, builds upon the strong foundation laid down by Collobert et al. (2011) for a neural network (NN) architecture and learning algorithm that can be applied to various natural language processing tasks, and ultimately is a variation of Nogueira dos Santos and Zadrozny (2014) who trained a NN for PoS tagging, with character-level and WE representations of words.

Also note that an earlier version of the system was used in Stemle (2016) to participate in the *EmpiriST 2015 shared task on automatic linguistic annotation of computer-mediated communication / social media* (Beißwenger et al., 2016).

2.1 Word Embeddings

Recently, state-of-the-art results on various linguistic tasks were accomplished by architectures using neural-network based WEs. Baroni et al. (2014) conducted a set of experiments comparing the popular `w2v` (Mikolov et al., 2013a; Mikolov et al., 2013b) implementation for creating WEs to other distributional methods with state-of-the-art results across various (semantic) tasks. These results suggest that the word embeddings substantially outperform the other architectures on semantic similarity and analogy detection tasks. Subsequently, Levy et al. (2015) conducted a comprehensive set of experiments and comparisons that suggest that much of the improved results are due to the system design and parameter optimizations, rather than the selected method. They conclude that "there does not seem to be a consistent significant advantage to one approach over the other".

Word embeddings provide high-quality low dimensional vector representations of words from large corpora of unlabelled data, and the representations, typically computed using NNs, encode many linguistic regularities and patterns (Mikolov et al., 2013b).

2.2 Character-Level Sub-Word Information

The morphology of a word is opaque to WEs, and the relatedness of the meaning of a lemma's different word forms, i.e. its different string representations, is *not* systematically encoded. This means that in morphologically rich languages with long-tailed frequency distributions, even some WE representations for word forms of common lemmata may become very poor (Kim et al., 2015).

We agree with Nogueira dos Santos and Zadrozny (2014) and Kim et al. (2015) that sub-word information is very important for PoS tagging, and therefore we augment the WE representations with character-level representations of the word beginnings and endings; thereby, we also stay language agnostic—at least, as much as possible—by avoiding the need for, often language specific, morphological pre-processing.

2.3 Recurrent Neural Network Layer

Language Models are a central part of NLP. They are used to place distributions over word sequences that encode systematic structural properties of the sample of linguistic content they are built from, and can then be used on novel content, e.g. to rank it or predict some feature on it. For a detailed overview on language modelling research see Mikolov (2012).

A straight-forward approach to incorporate WEs into feature-based language models is to use the embeddings' vector representations as features.[2] Having said that, WEs are also used in NN

[1] http://www.evalita.it/2016/tasks/postwita

[2] For an overview see, e.g. Turian et al. (2010).

architectures, where they constitute (part of) the input to the network.

Neural networks consist of a large number of simple, highly interconnected processing nodes in an architecture loosely inspired by the structure of the cerebral cortex of the brain (O'Reilly and Munakata, 2000). The nodes receive weighted inputs through these connections and *fire* according to their individual thresholds of their shared activation function. A firing node passes on an activation to all successive connected nodes. During learning the input is propagated through the network and the output is compared to the desired output. Then, the weights of the connections (and the thresholds) are adjusted step-wise so as to more closely resemble a configuration that would produce the desired output. After all input cases have been presented, the process typically starts over again, and the output values will usually be closer to the correct values.

RNNs are NNs where the connections between the elements are directed cycles, i.e. the networks have loops, and this enables them to model sequential dependencies of the input. However, regular RNNs have fundamental difficulties learning long-term dependencies, and special kinds of RNNs need to be used (Hochreiter, 1991); a very popular kind is the so called long short-term memory (LSTM) network proposed by Hochreiter and Schmidhuber (1997).

Overall, with this design we not only benefit from available labelled data but also from available general or domain-specific unlabelled data.

3 Implementation

We maintain the implementation in a source code repository at `https://github.com/bot-zen/`. The version tagged as `1.1` comprises the version that was used to generate the results submitted to the shared task (ST).

Our system feeds WEs and character-level subword information into a single-layer RNN with a LSTM architecture.

3.1 Word Embeddings

When computing WEs we take into consideration Levy et al. (2015): they observed that one specific configuration of `w2v`, namely the skip-gram model with negative sampling (SGNS) "is a robust baseline. While it might not be the best method for every task, it does not significantly underperform in any scenario. Moreover, SGNS is the fastest method to train, and cheapest (by far) in terms of disk space and memory consumption". Coincidentally, Mikolov et al. (2013b) also suggest to use SGNS. We incorporate `w2v`'s original C implementation for learning WEs[3] in an independent pre-processing step, i.e. we pre-compute the WEs. Then, we use gensim[4], a Python tool for unsupervised semantic modelling from plain text, to load the pre-computed data, and to compute the vector representations of input words for our NN.

3.2 Character-Level Sub-Word Information

Our implementation uses a *one-hot encoding* with a few additional features for representing subword information. The one-hot encoding transforms a categorical feature into a vector where the categories are represented by equally many dimensions with binary values. We convert a letter to lower-case and use the sets of ASCII characters, digits, and punctuation marks as categories for the encoding. Then, we add dimensions to represent more binary features like *'uppercase'* (was uppercase prior to conversion), *'digit'* (is digit), *'punctuation'* (is punctuation mark), *whitespace* (is white space, except the new line character; note that this category is usually empty, because we expect our tokens to *not* include white space characters), and *unknown* (other characters, e.g. diacritics). This results in vectors with more than a single *one-hot* dimension.

3.3 Recurrent Neural Network Layer

Our implementation uses Keras, a high-level NNs library, written in Python and capable of running on top of either TensorFlow or Theano (Chollet, 2015). In our case it runs on top of Theano, a Python library that allows to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently (The Theano Development Team et al., 2016).

The input to our network are sequences of the same length as the sentences we process. During training, we group sentences of the same length into batches and process the batches according to sentence length in increasing order. Each single word in the sequence is represented by its subword information and two WEs that come from two sources (see Section 4). For unknown words,

[3] `https://code.google.com/archive/p/word2vec/`

[4] `https://radimrehurek.com/gensim/`

i.e. words without a pre-computed WE, we first try to find the most similar WE considering 10 surrounding words. If this fails, the unknown word is mapped to a randomly generated vector representation. In Total, each word is represented by $2,280$ features: two times 500 (WEs), and sixteen times 80 for two 8-grams (word beginning and ending). If words are shorter than 8 characters their 8-grams are zero-padded.

This sequential input is fed into a LSTM layer that, in turn, projects to a fully connected output layer with softmax activation function. During training we use dropout for the projection into the output layer, i.e. we set a fraction (0.5) of the input units to 0 at each update, which helps prevent overfitting (Srivastava et al., 2014). We use categorical cross-entropy as loss function and backpropagation in conjunction with the RMSprop optimization for learning. At the time of writing, this was the Keras default—or the explicitly documented option to be used—for our type of architecture.

4 Results

We used our slightly modified implementation to participate in the *POS tagging for Italian Social Media Texts* (PoSTWITA) shared task (ST) of the 5th periodic evaluation campaign of Natural Language Processing (NLP) and speech tools for the Italian language EVALITA 2016. First, we describe the corpora used for training, and then the specific system configuration(s) for the ST.

4.1 Training Data for `w2v` and PoS Tagging

4.1.1 DiDi-IT (PoS, `w2v`)

didi-it (Frey et al., 2016) (version September 2016) is the Italian sub-part of the DiDi corpus, a corpus of South Tyrolean German and Italian from Facebook (FB) users' wall posts, comments on wall posts and private messages.

The Italian part consists of around 100,000 tokens collected from 20 profiles of Facebook users residing in South Tyrol. This version has about 20,000 PoS tags semi-automatically corrected by a single annotator.

The anonymised corpus is freely available for research purposes.

4.1.2 Italian UD (PoS, `w2v`)

Universal Dependencies (UD) is a project that is developing cross-linguistically consistent treebank annotation for many languages.[5]

italian-UD[6] (version from January 2015) corpus was originally obtained by conversion from ISDT (Italian Stanford Dependency Treebank) and released for the dependency parsing ST of EVALITA 2014 (Bosco et al., 2014). The corpus has semi-automatically converted PoS tags from the original two Italian treebanks, differing both in corpus composition and adopted annotation schemes.

The corpus contains around 317,000 tokens in around 13,000 sentences from different sources and genres. It is available under the CC BY-NC-SA 3.0[7] license.

4.1.3 PoSTWITA (PoS and `w2v`)

postwita is the Twitter data made available by the organizers of the ST. It contains Twitter tweets from the EVALITA2014 SENTIPLOC corpus: the development and test set and additional tweets from the same period of time were manually annotated for a global amount of 6438 tweets (114,967 tokens) and were distributed as the development set. The data is PoS tagged according to UD but with the additional insertion of seven Twitter-specific tags. All the annotations were carried out by three different annotators. The data was only distributed to the task participants.

4.1.4 C4Corpus (`w2v`)

c4corpus[8] is a full documents Italian Web corpus that has been extracted from CommonCrawl, the largest publicly available general Web crawl to date. See Habernal (2016) for details about the corpus construction pipeline, and other information about the corpus.

The corpus contains about $670m$ tokens in $22m$ sentences. The data is available under the CreativeCommons license family.

4.1.5 PAISÀ (`w2v`)

paisa (Lyding et al., 2014) is a corpus of authentic contemporary Italian texts from the web (harvested in September/October 2010). It was created

[5] `http://universaldependencies.org/`

[6] `http://universaldependencies.org/it/overview/introduction.html`

[7] Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported, i.e. the data can be copied and redistributed, and adapted for purposes other than commercial ones. See `https://creativecommons.org/licenses/by-nc-sa/3.0/` for more details.

[8] `https://github.com/dkpro/dkpro-c4corpus`

in the context of the project PAISÀ (Píattaforma per l'Apprendimento dell'Italiano Su corpora Annotati) with the aim to provide a large resource of freely available Italian texts for language learning by studying authentic text materials.

The corpus contains about $270m$ tokens in about $8m$ sentences. The data is available under the CC BY-NC-SA 3.0[9] license.

4.2 PoSTWITA shared task

For the ST we used one overall configuration for the system but three different corpus configurations for training. However, only one corpus configuration was entered into the ST: we used PoS tags from *didi-it + postwita* (run 1), from *italian-UD* (run 2), and from both (run 3). For `w2v` we trained a 500-dimensional skip-gram model on *didi-it + italian-UD + postwita* that ignored all words with less than 2 occurrences within a window size of 10; it was trained with negative sampling (value 15). We also trained a 500-dimensional skip-gram model on *c4corpus + paisa* that ignored all words with less than 33 occurrences within a window size of 10; it was trained with negative sampling (value 15).

The other `w2v` parameters were left at their default settings[10].

The evaluation of the systems was done by the organisers on unlabelled but pre-tokenised data (4759 tokens in 301 tweets), and was based on a token-by-token comparison. The considered metric was accuracy, i.e. the number of correctly assigned PoS tags divided by the total number of tokens.

(1) *didi-it + postwita*	**76.00**
(2) *italian-UD*	80.54
(3) *didi-it + postwita + italian-UD*	81.61
Winning Team	93.19

Table 1: Official result(s) of our PoS tagger for the three runs on the PoSTWITA ST data.

We believe, the unexpectedly little performance gain from utilizing the much larger *italian-UD* data over the rather small *didi-it + postwita* data may be rooted in the insertion of Twitter-specific tags into the data (see 4.1.3), something we did not account for, i.e. $18,213$ of $289,416$ and more importantly $7,778$ of $12,677$ sentences had imperfect information during training.

5 Conclusion & Outlook

We presented our submission to the PoSTWITA task of EVALITA 2016, where we participated with moderate results. In the future, we will try to rerun the experiment with training data that takes into consideration the Twitter-specific tags of the task.

Acknowledgments

The computational results presented have been achieved in part using the Vienna Scientific Cluster (VSC).

References

Jannis K. Androutsopoulos. 2007. Neue Medien – neue Schriftlichkeit? *Mitteilungen des Deutschen Germanistenverbandes*, 1:72–97.

Timothy Baldwin, Paul Cook, Marco Lui, Andrew MacKinlay, and Li Wang. 2013. How noisy social media text, how diffrnt social media sources? In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 356–364, Nagoya, Japan, October. Asian Federation of Natural Language Processing. `http://aclweb.org/anthology/I13-1041`.

Marco Baroni, Georgiana Dinu, and German Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247. Association for Computational Linguistics. `http://www.aclweb.org/anthology/P14-1023`.

Michael Beißwenger, Sabine Bartsch, Stefan Evert, and Kay-Michael Würzner. 2016. EmpiriST 2015: A Shared Task on the Automatic Linguistic Annotation of Computer-Mediated Communication, Social Media and Web Corpora. In *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*, pages 78–90, Berlin, Germany. Association for Computational Linguistics.

Michael Beißwenger. 2013. Das Dortmunder Chat-Korpus: ein annotiertes Korpus zur Sprachverwendung und sprachlichen Variation in der deutschsprachigen Chat-Kommunikation. *LINSE - Linguistik Server Essen*, pages 1–13.

Julian Benello, Andrew W. Mackie, and James A. Anderson. 1989. Syntactic category disambiguation with neural networks. *Computer Speech & Language*, 3(3):203–217, July. `http:`

[9] `https://creativecommons.org/licenses/by-nc-sa/3.0/`

[10] `-sample 1e-3 -iter 5 -alpha 0.025`

//www.sciencedirect.com/science/article/pii/0885230889900181.

Silvia Bernardini, Marco Baroni, and Stefan Evert. 2008. A WaCky Introduction. In *Wacky! Working papers on the Web as Corpus*, pages 9–40. GEDIT, Bologna, Italy. http://wackybook.sslmit.unibo.it/pdfs/bernardini.pdf.

Cristina Bosco, Felice Dell'Orletta, Simonetta Montemagni, Manuela Sanguinetti, and Maria Simi. 2014. The EVALITA 2014 Dependency Parsing Task. In *Proceedings of CLiC-it 2014 and EVALITA 2014*, pages 1–8. Pisa University Press.

Franois Chollet. 2015. Keras: Deep Learning library for Theano and TensorFlow. https://github.com/fchollet/keras.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537. https://arxiv.org/abs/1103.0398.

Jennifer-Carmen Frey, Aivars Glaznieks, and Egon W. Stemle. 2016. The DiDi Corpus of South Tyrolean CMC Data: A multilingual corpus of Facebook texts. Upcoming.

Eugenie Giesbrecht and Stefan Evert. 2009. Is Part-of-Speech Tagging a Solved Task? An Evaluation of POS Taggers for the German Web as Corpus. *Web as Corpus Workshop (WAC5)*. http://sigwac.org.uk/raw-attachment/wiki/WAC5/WAC5_proceedings.pdf#page=27.

Ivan Habernal, Omnia Zayed, and Iryna Gurevych. 2016. C4Corpus: Multilingual Web-size corpus with free license. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, page (to appear), Portorož, Slovenia, May. European Language Resources Association (ELRA).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Sepp Hochreiter. 1991. *Untersuchungen zu dynamischen neuronalen Netzen*. diploma thesis, TU München.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2015. Character-Aware Neural Language Models. *CoRR*, abs/1508.0. http://arxiv.org/abs/1508.06615.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225. https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/570.

Verena Lyding, Egon Stemle, Claudia Borghetti, Marco Brunello, Sara Castagnoli, Felice Dell Orletta, Henrik Dittmann, Alessandro Lenci, and Vito Pirrelli. 2014. The PAISÀ Corpus of Italian Web Texts. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, pages 36–43, Gothenburg, Sweden. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781. http://arxiv.org/abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. *CoRR*, abs/1310.4546, October. http://arxiv.org/abs/1310.4546.

Tomáš Mikolov. 2012. *Statistical Language Models Based on Neural Networks*. Ph.D. thesis, Brno University of Technology. http://www.fit.vutbr.cz/~imikolov/rnnlm/thesis.pdf.

Cícero Nogueira dos Santos and Bianca Zadrozny. 2014. Learning Character-level Representations for Part-of-Speech Tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826. http://jmlr.org/proceedings/papers/v32/santos14.pdf.

Randall C. O'Reilly and Yuko Munakata. 2000. *Computational Explorations in Cognitive Neuroscience Understanding the Mind by Simulating the Brain*. MIT Press. http://books.google.com/books?id=BLf34BFTaIUC{\&}pgis=1.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*, 15:1929–1958.

Egon W. Stemle. 2016. bot.zen @ EmpiriST 2015 - A minimally-deep learning PoS-tagger (trained for German CMC and Web data). In *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*, pages 115–119. Association for Computational Linguistics.

The Theano Development Team, Rami Al-Rfou, Guillaume Alain, Amjad Almahairi, and et al. 2016. Theano: A Python framework for fast computation of mathematical expressions. *CoRR*, abs/1605.02688. http://arxiv.org/abs/1605.02688.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=1858681.1858721.

A BiLSTM-CRF PoS-tagger for Italian tweets using morphological information

Fabio Tamburini
FICLIT - University of Bologna, Italy
fabio.tamburini@unibo.it

Abstract

English. This paper presents some experiments for the construction of an high-performance PoS-tagger for Italian using deep neural networks techniques (DNN) integrated with an Italian powerful morphological analyser that has been applied to tag Italian tweets. The proposed system ranked third at the EVALITA2016-PoSTWITA campaign.

Italiano. *Questo contributo presenta alcuni esperimenti per la costruzione di un PoS-tagger ad alte prestazioni per l'italiano utilizzando reti neurali 'deep' integrate con un potente analizzatore morfologico che è stato applicato all'annotazione di tweet. Il sistema si è classificato terzo nella campagna di valutazione EVALITA2016-PoSTWITA.*

1 Introduction

In recent years there were a large number of works trying to push the accuracy of the PoS-tagging task forward using new techniques, mainly from the deep learning domain (Collobert et al., 2011; Søgaard, 2011; dos Santos and Zadrozny, 2014; Huang et al., 2015; Wang et al., 2015; Chiu and Nichols, 2016).

In this study, still work-in-progress, we set-up a PoS-tagger for Italian able to gather the highest classification performances by using any available language resource and the most up-to-date DNN. We used AnIta (Tamburini and Melandri, 2012), one of the most powerful morphological analysers for Italian, based on a wide lexicon (about 110.000 lemmas), for providing the PoS-tagger with a large set of useful information.

The general PoS-tagger has been described in (Tamburini, 2016). This paper briefly describes the adaptation process we made for annotating Italian tweets.

2 Input features

The set of input features for each token is basically formed by two different components: the word embedding and some morphological information.

2.1 Word Embeddings

All the embeddings used in our experiments were extracted from a twitter corpus composed by 200 millions of tokens, belonging to 11 millions of tweets downloaded at the beginning of 2012 (February and March), by using the tool `word2vec`[1] (Mikolov et al., 2013). We added two special tokens to mark the sentence beginning '<s>' and ending '</s>'.

2.2 Morphological features, Unknown words handling and Sentence padding

As described in (Tamburini, 2016), we extended the word embeddings computed in a completely unsupervised way by concatenating to them a vector containing the possible PoS-tags provided by the AnIta analyser. This tool is also able to identify, through the use of simple regular expressions, numbers, dates, URLs, emails, etc., and to assign them the proper tag(s).

With regard to unknown words handling and sentence padding we followed the same procedure for the general tagger described in the cited paper, managing each sentence as one single sequence padded at the borders.

3 (Deep) Learning Blocks

All the experiments presented in this paper has been performed using Keras[2]. Keras provides some basic neural network blocks as well as different learning procedures for the desired network

[1] https://code.google.com/archive/p/word2vec/
[2] https://github.com/fchollet/keras/tree/master/keras

configuration and simple tools for writing new blocks. In our experiments we used Bidirectional Long Short-Term Memory - LSTM (Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005), and a new layer we wrote to handle Conditional Random Fields (CRF). We did some experiments stacking them after the softmax layer.

Figure 1 shows the DNN structure used in our experiments.

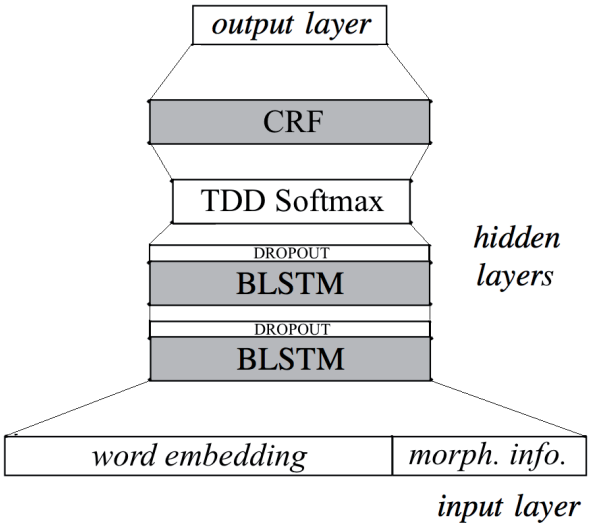


Figure 1: The DNN used in our experiments.

4 Experiments

During the set up phase, we did a lot of experiments for tuning the PoS-tagger using the Development Set. The following section describes the setup and the obtained results.

4.1 Hyper-Parameters

As for the general tagger (Tamburini, 2016), we did not test all the possible combinations; we used, instead, the most common set-up of parameters gathered from the literature. Table 1 outlines the whole setup for the unmodified hyper-parameters.

The DNN hidden layers were composed by 256 neurons.

4.2 The Early Stopping procedure

The usual way to set up an experiment following this suggestions involves splitting the gold standard into three different instance sets: the training set, for training, the validation set, to determine the stopping point, and the test set to evaluate the system. However, we are testing our systems on real evaluation data that has been already

`word2vec` **Embed.**		**Feature extraction**	
Hyperpar.	**Value**	**Hyperpar.**	**Value**
type	SkipGr.	window	5
size	100	**Learning Params.**	
(1/2) win.	5	batch (win)	1/4*NU
neg. sampl.	25	batch (seq)	1
sample	1e-4	Opt. Alg.	`Adam`
iter	15	Loss Func.	Categ.CE

Table 1: Unmodified hyper-parameters and algorithms used in our experiments. NU means the number of hidden or LSTM units per layer (the same for all layers). For `Adam` refer to (Kingma and Ba, 2015).

split by the organisers into development and test set. Thus, we can divide the development set into training/validation set for optimising the hyper-parameters and define the stopping epoch, but, for the final evaluation, we would like to train the final system on the complete development set to adhere to the evaluation constraints and to benefit from using more training data.

Having two different training procedures for the optimisation and evaluation phases leads to a more complex procedure for determining the stopping epoch. Moreover, the typical accuracy profile for DNN systems is not smooth and oscillate heavily during training. To avoid any problem in determining the stopping point we smoothed all the profiles using a bezier spline. The procedure we adopted to determine the stopping epoch is (please look at Fig. 2): (1) find the first maximum in the validation smoothed profile - A; (2) find the corresponding value of accuracy on the smoothed training profile - B; (3) find the point in the smoothed development set profile having the same accuracy as in B - C; (4) select the epoch corresponding at point C as the stopping epoch - D.

4.3 Results

First of all we split the Development Set into a proper training set (109,273 tokens) and a validation set (12,132 tokens) for setting up the entire system, to verify the correctness of the whole tagging process and to derive a first estimate of the tagger performances. We ran some experiments with three different seeds and, after having applied the early stop procedure described above, we derived the optimal stopping epoch to be used for the final testing and the tagging performances on the

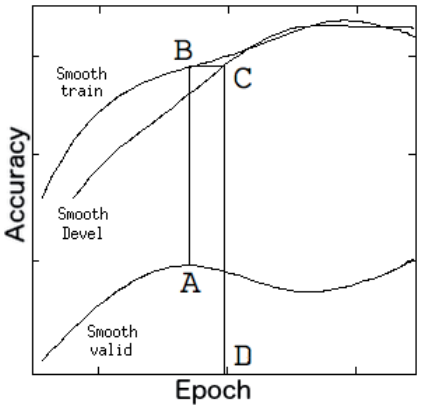


Figure 2: The early stopping procedure.

training/validation pair. Table 2 outlines these results.

	Tagging Accuracy	Stopping epoch
(A)	95.56	12
(B)	95.49	13
(C)	95.53	10
Avg.	95.53	

Table 2: The tagging results obtained in the setup phase.

We presented two kinds of results for the final evaluation: (1) the first official run was derived by applying the same random seed as the configuration (A), and (2) we submitted also, as an unofficial run, a tagged version obtained by combining all the three configurations using a voting scheme.

In Table 3 we can see our system performances, namely AnIta-BiLSTM-CRF (ABC), compared with all the systems that participated at the PoSTWITA 2016 task.

5 Conclusions and discussion

The proposed system for PoS-tagging, integrating DNNs and a powerful morphological analyser, exhibited very good accuracy results when applied to the PoSTWITA task of the EVALITA 2016 campaign.

Looking at the official results, and comparing them with the experiments we devised to set up our system, it is easy to note the large difference in performances. During the setup phase we obtained coherent results well above 95% of accuracy, while the best performing system in the official evaluation exhibit performances slightly above 93%. It is a huge difference for this kind of task, rarely observed in real experiments.

#	TEAM	Tagging Accuracy
1	Team1	0.931918 (4435/4759)
2	Team2	0.928556 (4419/4759)
3	**ABC_UnOFF**	0.927926 (4416/4759)
4	Team4	0.927086 (4412/4759)
5	**ABC**	0.924564 (4400/4759)
6	Team5	0.922463 (4390/4759)
7	Team5_UnOFF	0.918470 (4371/4759)
8	Team6	0.915739 (4358/4759)
9	Team6_UnOFF	0.915318 (4356/4759)
10	Team7	0.878966 (4183/4759)
11	Team8	0.859634 (4091/4759)
12	Team2_UnOFF	0.817819 (3892/4759)
13	Team9	0.760034 (3617/4759)

Table 3: EVALITA2016 - PoSTWITA participants' results with respect to Tagging Accuracy. "UnOFF" marks unofficial results.

In my opinion there is only one reason that explains this difference in performances: the documents in the test set are not drawn from the same kind of corpus as the development set and this is not a desirable condition unless you explicitly organise a domain adaptation task. The TS, as well as the DS, have been inherited from the SENTIPOLC task of the same evaluation campaign, thus the problem could be the same also for other tasks of the same evaluation campaign.

References

Jason Chiu and Eric Nichols. 2016. Sequential Labeling with Bidirectional LSTM-CNNs. In *Proc. International Conf. of Japanese Association for NLP*, pages 937–940.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Cicero dos Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proc. of the 31st International Conference on Machine Learning, JMLR*, volume 32. JMLR W&CP.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5-6):602–610.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *ArXiv e-prints, 1508.01991*.

D.P. Kingma and J.L. Ba. 2015. Adam: a method for stochastic optimization. In *Proc. International Conference on Learning Representations - ICLR.*, pages 1–13.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proc. of Workshop at ICLR*.

Anders Søgaard. 2011. Semi-supervised condensed nearest neighbor for part-of-speech tagging. In *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 48–52, Portland, Oregon, USA.

Fabio Tamburini and Matias Melandri. 2012. AnIta: a powerful morphological analyser for Italian. In *Proc. 8th International Conference on Language Resources and Evaluation - LREC 2012*, pages 941–947, Istanbul.

Fabio Tamburini. 2016. (Better than) State-of-the-Art PoS-tagging for Italian Texts. In *Proc. Third Italian Conference on Computational Linguistics - CLiC-it*, Napoli.

Peilu Wang, Yao Qian, Frank. K Soong, Lei He, and Hai Zhao. 2015. A Unified Tagging Solution: Bidirectional LSTM Recurrent Neural Network with Word Embedding. *ArXiv e-prints, 1511.00215*.

Overview of the EVALITA 2016 Question Answering for Frequently Asked Questions (QA4FAQ) Task

Annalina Caputo[1] and **Marco de Gemmis**[2,5] and **Pasquale Lops**[2]

Francesco Lovecchio[3] and **Vito Manzari**[4]

[1] ADAPT Centre, Dublin

[2] Department of Computer Science, University of Bari Aldo Moro

[3] Acquedotto Pugliese (AQP) S.p.a. [4] Sud Sistemi S.r.l. [5] QuestionCube S.r.l.

[1]annalina.caputo@adaptcentre.ie

[2]{marco.degemmis,pasquale.lops}@uniba.it

[3]f.lovecchio@aqp.it [4]manzariv@sudsistemi.it

[5]marco.degemmis@questioncube.com

Abstract

English. This paper describes the first edition of the Question Answering for Frequently Asked Questions (QA4FAQ) task at the EVALITA 2016 campaign. The task concerns the retrieval of relevant frequently asked questions, given a user query. The main objective of the task is the evaluation of both question answering and information retrieval systems in this particular setting in which the document collection is composed of FAQs. The data used for the task are collected in a real scenario by AQP Risponde, a semantic retrieval engine used by Acquedotto Pugliese (AQP, the Organization for the management of the public water in the South of Italy) for supporting their customer care. The system is developed by QuestionCube, an Italian startup company which designs Question Answering tools.

Italiano. *Questo lavoro descrive la prima edizione del Question Answering for Frequently Asked Questions (QA4FAQ) task proposto durante la campagna di valutazione EVALITA 2016. Il task consiste nel recuperare le domande più frequenti rilevanti rispetto ad una domanda posta dall'utente. L'obiettivo principale del task è la valutazione di sistemi di question answering e di recupero dell'informazione in un contesto applicativo reale, utilizzando i dati provenienti da AQP Risponde, un motore di ricerca semantico usato da Acquedotto Pugliese (AQP, l'ente per la gestione dell'acqua pubblica nel Sud Italia). Il sistema è sviluppato da QuestionCube, una startup italiana che progetta soluzioni di Question Answering.*

1 Motivation

Searching within the Frequently Asked Questions (FAQ) page of a web site is a critical task: customers might feel overloaded by many irrelevant questions and become frustrated due to the difficulty in finding the FAQ suitable for their problems. Perhaps they are right there, but just worded in a different way than they know.

The proposed task consists in retrieving a list of relevant FAQs and corresponding answers related to the query issued by the user.

Acquedotto Pugliese (AQP) developed a semantic retrieval engine for FAQs, called AQP Risponde[1], based on Question Answering (QA) techniques. The system allows customers to ask their own questions, and retrieves a list of relevant FAQs and corresponding answers. Furthermore, customers can select one FAQ among those retrieved by the system and can provide their feedback about the perceived accuracy of the answer.

AQP Risponde poses relevant research challenges concerning both the usage of the Italian language in a deep QA architecture, and the variety of language expressions adopted by customers to formulate the same information need.

The proposed task is strongly related to the one recently organized at Semeval 2015 and 2016 about Answer Selection in Community Question Answering (Nakov et al., 2015). This task helps to automate the process of finding good answers to new questions in a community-created discussion forum (e.g., by retrieving similar questions in

[1]http://aqprisponde.aqp.it/ask.php

the forum and by identifying the posts in the answer threads of similar questions that answer the original one as well). Moreover, the QA-FAQ has some common points with the Textual Similarity task (Agirre et al., 2015) that received an increasing amount of attention in recent years.

The paper is organized as follows: Section 2 describes the task, while Section 3 provides details about competing systems. Results of the task are discussed in Section 4.

2 Task Description: Dataset, Evaluation Protocol and Measures

The task concerns the retrieval of relevant frequently asked questions, given a user query. For defining an evaluation protocol, we need a set of FAQs, a set of user questions and a set of relevance judgments for each question. In order to collect these data, we exploit an application called AQP Risponde, developed by QuestionCube for the Acquedotto Pugliese. AQP Risponde provides a back-end that allows to analyze both the query log and the customers' feedback to discover, for instance, new emerging problems that need to be encoded as FAQ. AQP Risponde is provided as web and mobile application for Android[2] and iOS[3] and is currently running in the Acquedotto Pugliese customer care. AQP received about 25,000 questions and collected about 2,500 user feedback. We rely on these data to build the dataset for the task. In particular, we provide:

- a knowledge base of 406 FAQs. Each FAQ is composed of a question, an answer, and a set of tags;
- a set of 1,132 user queries. The queries are collected by analyzing the AQP Risponde system log. From the initial set of queries, we removed queries that contains personal data;
- a set of 1,406 pairs $< query, relevant faq >$ that are exploited to evaluate the contestants. We build these pairs by analyzing the user feedback provided by real users of AQP Risponde. We manually check the user feedback in order to remove noisy or false feedback. The check was performed by two experts of the AQP customer support.

[2] https://play.google.com/store/apps/details?id=com.questioncube.aqprisponde&hl=it

[3] https://itunes.apple.com/it/app/aqp-risponde/id1006106860

We provided a little sample set for the system development and a test set for the evaluation. We did not provide a set of training data: AQP is interested in the development of unsupervised systems because AQP Risponde must be able to achieve good performance without any user feedback. Following, an example of FAQ is reported:

Question "Come posso telefonare al numero verde da un cellulare?" *How can I call the toll-free number by a mobile phone?*

Answer "È possibile chiamare il Contact Center AQP per segnalare un guasto o per un pronto intervento telefonando gratuitamente anche da cellulare al numero verde 800.735.735. Mentre per chiamare il Contact Center AQP per servizi commerciali 800.085.853 da un cellulare e dall'estero è necessario comporre il numero +39.080.5723498 (il costo della chiamata è secondo il piano tariffario del chiamante)." *You can call the AQP Contact Center to report a fault or an emergency call without charge by the phone toll-free number 800 735 735...*

Tags *canali, numero verde, cellulare*

For example, the previous FAQ is relevant for the query: "Si può telefonare da cellulare al numero verde?" *Is it possible to call the toll-free number by a mobile phone?*

Moreover, we provided a simple baseline based on a classical information retrieval model.

2.1 Data Format

FAQs are provided in both XML and CSV format using ";" as separator. The file is encoded in UTF-8 format. Each FAQ is described by the following fields:

id a number that uniquely identifies the FAQ

question the question text of the current FAQ

answer the answer text of the current FAQ

tag a set of tags separated by ","

Test data are provided as a text file composed by two strings separated by the *TAB* character. The first string is the user *query id*, while the second string is the text of the user query. For example: "1 Come posso telefonare al numero verde da un cellulare?" and "2 Come si effettua l'autolettura del contatore?".

2.2 Baseline

The baseline is built by using Apache Lucene (ver. 4.10.4)[4]. During the indexing for each FAQ, a document with four fields (*id, question, answer, tag*) is created. For searching, a query for each question is built taking into account all the question terms. Each field is boosted according to the following score *question=4*, *answer=2* and *tag=1*. For both indexing and search the *ItalianAnalyzer* is adopted. The top 25 documents for each query are provided as result set. The baseline is freely available on GitHub[5] and it was released to participants after the evaluation period.

2.3 Evaluation

The participants must provide results in a text file. For each query in the test data, the participants can provide 25 answers at the most, ranked according by their systems. Each line in the file must contain three values separated by the TAB character: $< queryid >< faqid >< score >$.

Systems are ranked according to the accuracy@1 (c@1). We compute the precision of the system by taking into account only the first correct answer. This metric is used for the final ranking of systems. In particular, we take into account also the number of unanswered questions, following the guidelines of the CLEF ResPubliQA Task (Peñas et al., 2009). The formulation of c@1 is:

$$c@1 = \frac{1}{n}(n_R + n_U \frac{n_R}{n}) \quad (1)$$

where n_R is the number of questions correctly answered, n_U is the number of questions unanswered, and n is the total number of questions.

The system should not provide result for a particular question when it is not confident about the correctness of its answer. The goal is to reduce the amount of incorrect responses, keeping the number of correct ones, by leaving some questions unanswered. Systems should ensure that only the portion of wrong answers is reduced, maintaining as high as possible the number of correct answers. Otherwise, the reduction in the number of correct answers is punished by the evaluation measure for both the answered and unanswered questions.

[4] `http://lucene.apache.org/`

[5] `https://github.com/swapUniba/qa4faq`

3 Systems

Thirteen teams registered in the task, but only three of them actually submitted the results for the evaluation. A short description of each system follows:

chiLab4It - The system described in (Pipitone et al., 2016a) is based on the cognitive model proposed in (Pipitone et al., 2016b). When a support text is provided for finding the correct answer, QuASIt is able to use this text to find the required information. ChiLab4It is an adaptation of this model to the context of FAQs, in this case the FAQ is exploited as support text: the most relevant FAQ will be the one whose text will best fit the user's question. The authors define three similarity measures for each field of the FAQ: question, answer and tags. Moreover, an expansion step by exploiting synonyms is applied to the query. The expansion module is based on Wiktionary.

fbk4faq - In (Fonseca et al., 2016), the authors proposed a system based on vector representations for each query, question and answer. Query and answer are ranked according to the cosine distance to the query. Vectors are built by exploring the word embeddings generated by (Dinu et al., 2014), and combined in a way to give more weight to more relevant words.

NLP-NITMZ the system proposed by (Bhardwaj et al., 2016) is based on a classical VSM model implemented in Apache Nutch[6]. Moreover, the authors add a combinatorial searching technique that produces a set of queries by several combinations of all the keywords occurring in the user query. A custom stop word list was developed for the task, which is freely available[7].

It is important to underline that all the systems adopt different strategies, while only one system (*chiLab4It*) is based on a typical question answer module. We provide a more detailed analysis about this aspect in Section 4.

Table 1: System results.

System	c@1
qa4faq16.chilab4it.01	0.4439
baseline	*0.4076*
qa4fac16.fbk4faq.2	0.3746
qa4fac16.fbk4faq.1	0.3587
qa4fac16.NLP-NITMZ.1	0.2125
qa4fac16.NLP-NITMZ.2	0.0168

4 Results

Results of the evaluation in terms of c@1 are reported in Table 1. The best performance is obtained by the *chilab4it* team, that is the only one able to outperform the baseline. Moreover, the *chilab4it* team is the only one that exploits question answering techniques: the good performance obtained by this team proves the effectiveness of question answering in the FAQ domain. All the other participants had results under the baseline. Another interesting outcome is that the baseline exploiting a simple VSM model achieved remarkable results.

A deep analysis of results is reported in (Fonseca et al., 2016), where the authors have built a custom development set by paraphrasing original questions or generating a new question (based on original FAQ answer), without considering the original FAQ question. The interesting result is that their system outperformed the baseline on the development set. The authors underline that the development set is completely different from the test set which contains sometime short queries and more realistic user's requests. This is an interesting point of view since one of the main challenge of our task concerns the variety of language expressions adopted by customers to formulate the information need. Moreover, in their report the authors provide some examples in which the FAQ reported in the gold standard is less relevant than the FAQ reported by their system, or in some cases the system returns a correct answer that is not annotated in the gold standard. Regarding the first point, we want to point out that our relevance judgments are computed according to the users' feedback and reflect their concept of relevance[8]. We tried to mitigate issues related to relevance judgments by manually checking users' feedback. However, this manual annotation process might have introduced some noise, which is common to all participants.

Regarding missing correct answers in the gold standard: this is a typical issue in the retrieval evaluation, since it is impossible to assess all the FAQ for each test query. Generally, this issue can be solved by creating a pool of results for each query. Such pool is built by exploiting the output of several systems. In this first edition of the task, we cannot rely on previous evaluations on the same set of data, therefore we chose to exploit users' feedback. In the next editions of the task, we can rely on previous results of participants to build that pool of results.

Finally, in Table 2 we report some information retrieval metrics for each system[9]. In particular, we compute Mean Average Precision (MAP), Geometrical-Mean Average Precision (GMAP), Mean Reciprocal Rank (MRR), Recall after five (R@5) and ten (R@10) retrieved documents. Finally we report the success_1 that is equal to c@1, but without taking into account answered queries. We can notice that on retrieval metrics the baseline is the best approach. This was quite expected since an information retrieval model tries to optimize retrieval performance. Conversely, the best approach according to success_1 is the *chilab4it* system based on question answering, since it tries to retrieve a correct answer in the first position. This result suggests that the most suitable strategy in this context is to adopt a question answering model, rather than to adapt an information retrieval approach. Another interesting outcome concerns the system *NLP-NITMZ.1*, which obtains an encouraging success_1, compared to the c@1. This behavior is ascribable to the fact that the system does not adopt a strategy that provides an answer for all queries.

5 Conclusions

For the first time for the Italian language, we propose a question answering task for frequently asked questions. Given a user query, the participants must provide a list of FAQs ranked by relevance according to the user need. The collection

[6] https://nutch.apache.org

[7] https://github.com/SRvSaha/QA4FAQ-EVALITA-16/blob/master/italian_stopwords.txt

[8] Relevance is subjective.

[9] Metrics are computed by the latest version of the trec_eval tool: http://trec.nist.gov/trec_eval/

Table 2: Results computed by using typical information retrieval metrics

System	MAP	GMAP	MRR	R@5	R@10	success_1
chilab4it	0.5149	0.0630	0.5424	0.6485	0.7343	0.4319
baseline	0.5190	0.1905	0.5422	0.6805	0.7898	0.4067
fbk4faq.2	0.4666	0.0964	0.4982	0.5917	0.7244	0.3750
fbk4faq.1	0.4473	0.0755	0.4781	0.5703	0.6994	0.3578
NLP-NITMZ.1	0.3936	0.0288	0.4203	0.5060	0.5879	0.3161
NLP-NITMZ.2	0.0782	0.0202	0.0799	0.0662	0.1224	0.0168

of FAQs was built by exploiting a real application developed by QuestionCube for Acquedotto Pugliese. The relevance judgments for the evaluation are built by taking into account the user feedback.

Results of the evaluation demonstrated that only the system based on question answering techniques is able to outperform the baseline, while all the other participants reported results under the baseline. Some issues pointed out by participants suggest exploring a pool of results for building more accurate judgments. We plan to implement this approach in future editions of the task.

Acknowledgments

This work is supported by the project "Multilingual Entity Liking" funded by the Apulia Region under the program FutureInResearch.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalara, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Divyanshu Bhardwaj, Partha Pakray, Jereemi Bentham, Saurav Saha, and Alexander Gelbukh. 2016. Question Answering System for Frequently Asked Questions. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. 2014. Improving zero-shot learning by mitigating the hubness problem. *arXiv preprint arXiv:1412.6568*.

Erick R. Fonseca, Simone Magnolini, Anna Feltracco, Mohammed R. H. Qwaider, and Bernardo Magnini. 2016. Tweaking Word Embeddings for FAQ Ranking. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Preslav Nakov, Lluıs Marquez, Walid Magdy, Alessandro Moschitti, James Glass, and Bilal Randeree. 2015. Semeval-2015 task 3: Answer selection in community question answering. *SemEval-2015*, page 269.

Anselmo Peñas, Pamela Forner, Richard Sutcliffe, Álvaro Rodrigo, Corina Forăscu, Iñaki Alegria, Danilo Giampiccolo, Nicolas Moreau, and Petya Osenova. 2009. Overview of ResPubliQA 2009: question answering evaluation over European legislation. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 174–196. Springer.

Arianna Pipitone, Giuseppe Tirone, and Roberto Pirrone. 2016a. ChiLab4It System in the QA4FAQ Competition. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Arianna Pipitone, Giuseppe Tirone, and Roberto Pirrone. 2016b. QuASIt: a Cognitive Inspired Approach to Question Answering System for the Italian Language. In *Proceedings of the 15th International Conference on the Italian Association for Artificial Intelligence 2016*. aAcademia University Press.

Question Answering System for Frequently Asked Questions

Divyanshu Bhardwaj, Partha Pakray, Jereemi Bentham, Saurav Saha
Dept. of CSE
NIT Mizoram
India
(divbhardwaj42, parthapakray, jereemibentham, contact.srvsaha)@gmail.com

Alexander Gelbukh
CIC
IPN Mexico
Mexico
gelbukh@gelbukh.com

Abstract

English. Question Answering (QA) is an important aspect of Natural Language Processing. It comprises building a system that automatically answers questions sought in natural language. Frequently Asked Questions (FAQs) are a set of listed questions and answers concerning a specific topic, which are most likely to be enquired by a user. This paper deals with developing an open domain QA system for retrieving a list of relevant FAQs related to the query issued by the user. Our approach combines the orthodox AND/OR searching with the Combinatorics searching technique which is able to produce an exhaustive list of results for a particular query generated.

Italiano. *Question Answering (QA) un aspetto importante di Natural Language Processing. Si compone di costruire un sistema che risponde automaticamente alle domande cercato in linguaggio naturale. Domande frequenti (FAQ) sono un insieme di domande elencate e risposte riguardanti un argomento specifico, che hanno pi probabilit di indagato da un utente. Questo documento si occupa di sviluppo di un sistema di QA dominio aperto per il recupero di un elenco di domande frequenti pertinenti relativi alla query emesso da parte dell'utente. Il nostro approccio combina l'ortodossa e / o la ricerca con la tecnica di ricerca combinatorio che in grado di produrre un elenco esaustivo dei risultati per una determinata query generato.*

1 Introduction

Question Answering (QA) is an emerging topic in today's world. It is an aggregate of Information Retrieval (IR) and Natural Language Processing (NLP) and is concerned with developing an automated engine which is able to respond to the queries presented by users in natural language.
Frequently Asked Questions (FAQs) represent an effective and efficient way to quickly resolve queries posed by users. They are usually represented as an ensembled list of questions and their answers.

Searching within FAQs can be a tedious task. This becomes even more drawn out when paraphrasing comes into fray. As a result the user is pushed into a maze of questions and answers having to manually look for a particular one as shown in figure 1. It is here that a QA system comes of utmost importance retrieving the particular desired query instantly.

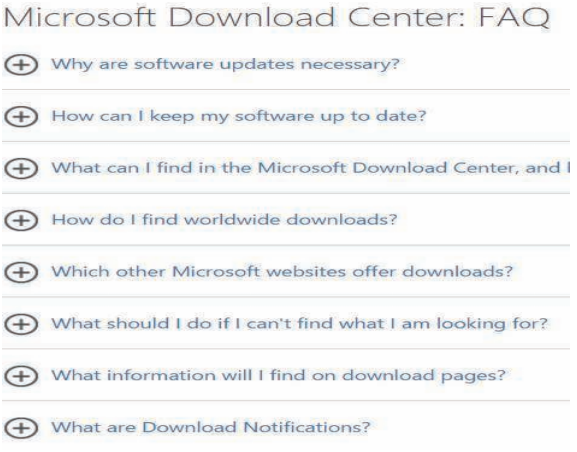


Figure 1: FAQs of Microsoft Download Center

The rest of this paper is organised as follows, Section 2 describes the corpus and its preprocessing, Section 3 describes our system's architecture and the tools used, Section 4 describes the experiment. Section 5 describes the performance of the system, Section 6 analyses the results and Section 7 describes the conclusion and future works.

2 Corpus and Preprocessing

The corpus obtained from the QA4FAQ task website[1] provided us with FAQ in .csv (comma separated values) format, using *;* as separator and in XML format. The CSV file was in UTF-8 format and contained 4 fields viz.,

1. *id:* a number that uniquely identifies the FAQ;
2. *question:* the question text of the current FAQ;
3. *answer:* the answer text of the current FAQ;
4. *tag:* a set of tags separated by ,.

An example of the data provided is given below:

193;Cosa significa AEEGSI?; l'Autorit per l'Energia Elettrica il Gas ed il Sistema Idrico.;acqua, acquedotto, distribuzione, AEEGSI

2.1 Parsing

For the purpose of pre-processing of the training data we developed a CSV parser which could extract the *ID* and the rest of the parts. Development dataset had 406 files with id, question, answer, tag(s). We extracted the question, answer and tags in a file and saved it in the file named *ID.txt*.

2.2 Stopword Removal

In order to increase the efficiency of our input data, we decided to perform stopwords removal. Words which occur in 80% of the documents in the collection are the stop words. However while searching for a list of Italian stopwords, we realised that the existing ones had only 133 to 399 stopwords.[2] [3] [4] So, we merged them and developed our own exhaustive Italian stopword corpus from the existing ones. This corpus[5] had approximately 546 unique stopwords in total. This operation helped us in getting rid of the unwanted words which would hinder the system's performance.

3 System Architecture

The architecture of our system is shown in figure 2.

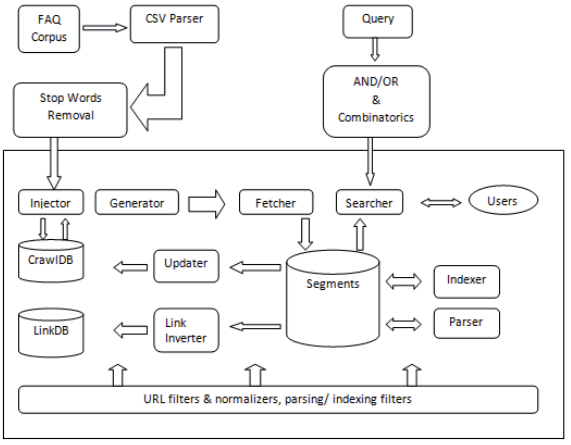


Figure 2: Architecture of the implemented system

The architecture may be divided into two distinct parts as shown in figure. One part contains the architecture of Nutch[6] enclosed in the rectangle. It contains all the basic components essential in the implementation of a Search Engine. The other part represents the aggregation of the searching techniques to be adopted while searching the FAQs. This includes a module that processes the queries obtained for both AND/OR searching as well as combinatorics based searching.The two major steps involved in developing the architecture were *Crawling & Indexing* and *Searching* (described in Section 4).
The steps involved in crawling and indexing are described below:

[1] http://qa4faq.github.io

[2] http://www.ranks.nl/stopwords/italian

[3] http://members.unine.ch/jacques.savoy/clef/italianST.txt

[4] https://github.com/themnd/stopword-it/blob/master/stopwords.txt

[5] The corpus is openly shared in Github for furthur use - https://github.com/SRvSaha/QA4FAQ-EVALITA-16/blob/master/italian_stopwords.txt

[6] https://nutch.apache.org

1. Run a generic Java code taking the ids (taken from *ID.txt*) as the input to generate URL seeds.
2. Injector injects the list of seed URLs into the crawlDB.
3. Generator takes the list of seed URLs from crawlDB, forms fetch list, adds crawl_generate folder into the segments.
4. These fetch lists are used by fetchers to fetch the raw content of the document. It is then stored in segments.
5. Parser is called to parse the content of the document and parsed content is stored back in segments.
6. The links are inverted in the link graph and stored in LinkDB.
7. Indexing the terms present in segments is done and indices are updated in the segments.
8. Information on the newly fetched documents are updated on the crawlDB.

4 Experiments

The corpus obtained after pre-processing was experimented upon by means of various methodologies. A total of 1132 FAQs were available in the test data set. A prototype system was created by feeding the input data into Nutch. We performed two separate runs so as to perform a comparative study between unprocessed and pre processed data.
We used Nutch's own configuration for the Indexing, Searching and Ranking of the data for one of the runs and implemented our own configuration for the other run. The ranking provided by Nutch may be explained using the following equation:

$$score(\vec{q}, \vec{d}) = queryNorm(\vec{q}) * coord(\vec{q}, \vec{d}) * norm(t, \vec{d}) * \sum_{t \in \vec{d}} (tf(t) * idf(t) * t.boost(t.field))$$

Figure 3: Nutch's Ranking Equation

Here,

1. *queryNorm() :* indicates the normalization factor for the query.
2. *coord() :* indicates how many query terms are present in the given document.
3. *norm() :* score indicating field based normalization factor.
4. *tf:* term frequency
5. *idf:* inverse document frequency
6. *t.boost() :* score indicating the importance of terms occurrence in a particular field

Apart from this, we developed our own configuration which was a combination of both the traditional AND/OR search along with the Combinatorics approach. To implement this Combinatorics approach, we split the query by space separator and all possible combinations of a word in query were generated. This is the methodology adopted in subset generation from a given set. So, given n number of words in a query after removing stopwords, we would have $2^n - 1$ possible combinations of query. These were then used for searching by Nutch and ranking was done based on the ranking algorithm we developed. Benefit of this approach was that, it was an exhaustive search and maximum number of relevant results would be retrieved using it using proper ranking algorithm.
This approach could be explained using the following example:
Consider the following query:
numero verde aqp
For this query, all the possible combinations would be created in the following order :
numero verde aqp
numero verde
verde aqp
numero aqp
numero
verde
aqp

From this example we can clearly visualize how this approach would be extremely efficient in retrieving the most relevant answers for queries provided by the user. After applying this approach, we were left with 29 unanswered queries.
We also implemented our own ranking system which ranked the retrieved pages in the following

way :
Consider a query of 4 words. We used a 4 point scale to rank the pages with the highest score being assigned to the page with *4*(number of matches)* Thus, for a query of length *n*, the highest match would be assigned to *n*(number of matches)*. Assuming we have a query of *n* words, all possible combinations i.e, $2^n - 1$ possible queries were to be ranked according to the above mentioned algorithm.
Consider the query following query:
numero verde
and let the text be *il numero verde non verde, un numero che pu essere dipinta di verde*.
Ranking of queries would be done as :
1. *numero verde* : 2*1 = 2
2. *numero* : 1*2 = 2
3. *verde* : 1*3 = 3
Since we get the highest score from the query *verde* so the most relevant document will be fetched by *verde*. Our system retrieved results based on this methodology.

5 Performance

The relevant statistics of both the runs based on the experiments performed are outlined in Table 1.

Run 1		
Total no. of queries	No. of queries answered	No. of queries unanswered
1132	684	448
Run 2		
Total no. of queries	No. of queries answered	No. of queries unanswered
1132	1103	29

Table 1: Statistics of both approaches

As can be inferred from Table 1, while during Run 1 there were a large number of unanswered queries, they were significantly reduced in Run 2. This was possible due to the combinatorics approach used in Run 2. The performance of our system in both the runs is depicted in Table 2.

Runs	Score Obtained
Run 1	0.2125
Run 2	0.0168

Table 2: Performance of NLP-NITMZ in both runs

Systems were ranked according to *accuracy@1*. In this method of ranking the precision of the system was computed taking into account only the first answer generated by the system. The formulation of *c@1* is given as below:

$$c@1 = \frac{1}{n}(n_R + n_U \frac{n_R}{n})$$

Figure 4: Formula for c@1

where:

1. n_R : number of questions correctly answered
2. n_R : number of questions unanswered
3. n: total number of questions

6 Discussion

As the evaluation was done according to *accuracy@1* which considered only the first answer retrieved by the systems, the results obtained weren't extremely accurate. We however managed to implement a search engine which was 97.33% accurate in retrieving queries, which resulted in a trivial amount of unanswered queries. This system conveyed a lot of information which made us realise that combinatorics can be an extremely powerful tool for searching if implemented in a proper way. However, the relevancy of the results obtained would depend on how efficiently the ranking is done.

7 Conclusion and Future Direction

In this paper, we intended to frame an automated Question Answering (QA) system for Frequently Asked Questions (FAQs). We described the preprocessing of the corpus and the experiments performed on them. We also described the combinatorics approach used for searching. While the evaluation results were only decent, we did manage to materialise a remarkably accurate search engine for FAQs. Now that we have an adept search engine we would next endeavour towards perfecting our ranking techniques and algorithms in order to take steps towards implementing a state of the art QA system.

Acknowledgments

This work presented here falls under the research project Grant No. YSS/2015/000988 and supported by the Department of Science & Technology (DST) and Science and Engineering Research Board (SERB), Govt. of India. The authors would like to acknowledge the Department of Computer Science & Engineering, National Institute of Technology, Mizoram for providing infrastructural facilities in order to facilitate research on this task.

References

Annalina Caputo, Marco de Gemmis, Pasquale Lops, Franco Lovecchio and Vito Manzari 2016. *Overview of the EVALITA 2016 Question Answering for Frequently Asked Questions (QA4FAQ) Task*, Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016). aAcademia University Press

Deepak Ravichandran and Eduard Hovy 2002. *Learning Surface Text Patterns for a Question Answering System*, Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)

Lynette Hirschman and Robert Gaizauskas 2001. *Natural language question answering: the view from here*, Natural Language Engineering

Marius Pasca and Sanda Harabagiu 2001. *High Performance Question/Answering*, ACM SIGIR-2001

Narendra K Gupta, Mazin G Rahim, Giuseppe and Riccardi, 2007. *System for handling frequently asked questions in a natural language dialog service*, Google Patents

Partha Pakray 2014. *Yamraj: Binary-class and Multi-class based Textual Entailment System for Japanese (JA) and Chinese Simplified (CS)*, Proceedings of the 11th NTCIR Conference

Partha Pakray and Petr Sojka 2014. *An Architecture for Scientific Document Retrieval Using Textual and Math Entailment Modules*, Recent Advances in Slavonic Natural Language Processing, Karlova Studnka, Czech Republic

Partha Pakray, Pinaki Bhaskar, Somnath Banerjee, Bidhan Chandra Pal, Sivaji Bandyopadhyay and Alexander Gelbukh 2011. *A Hybrid Question Answering System based on Information Retrieval and Answer Validation*, CLEF 2011 Workshop on Question Answering For Machine Reading Evaluation (QA4MRE), CLEF 2011 Labs and Workshop

Pinaki Bhaskar, Amitava Das, Partha Pakray and Sivaji Bandyopadhyay 2010. *Theme Based English and Bengali Ad-hoc Monolingual Information Retrieval in FIRE 2010*, FIRE 2010

Partha Pakray, Pinaki Bhaskar, Santanu Pal, Dipankar Das, Sivaji Bandyopadhyay and Alexander Gelbukh 2010. *JU_CSE_TE: System Description QA@CLEF 2010 - ResPubliQA*, CLEF 2010 Workshop on Multiple Language Question Answering (MLQA 2010)

Pinaki Bhaskar, Partha Pakray, Somnath Banerjee, Samadrita Banerjee, Sivaji Bandyopadhyay and Alexander F Gelbukh 2012. *Question Answering System for QA4MRE@ CLEF 2012*, CLEF (Online Working Notes/Labs/Workshop)

Pinaki Bhaskar, Partha Pakray, Somnath Banerjee, Samadrita Banerjee, Sivaji Bandyopadhyay and Alexander Gelbukh 2012. *Question Answering System for QA4MRE@CLEF 2012*, Workshop on Question Answering For Machine Reading Evaluation (QA4MRE), CLEF 2012 Labs and Workshop

Pinaki Bhaskar, Somnath Banerjee, Partha Pakray, Samadrita Banerjee, Sivaji Bandyopadhyay and Alexander F Gelbukh 2013. *A hybrid question answering system for Multiple Choice Question (MCQ)*, CEUR-WS

Robin D Burke, Kristian J Hammond, Vladimir Kulyukin, Steven L Lytinen, Noriko Tomuro, and Scott Schoenberg 1997. *Question answering from frequently asked question files: Experiences with the faq finder system*, AI magazine

Somnath Banerjee, Pinaki Bhaskar, Partha Pakray, Sivaji Bandyopadhyay and Alexander F Gelbukh 2013. *Multiple Choice Question (MCQ) Answering System for Entrance Examination*, CLEF (Working Notes)

Valentin Jijkoun and Maarten de Rijke 2010. *Retrieving answers from frequently asked questions pages on the web*, Proceedings of the 14th ACM international conference on Information and knowledge management

Tweaking Word Embeddings for FAQ Ranking

Erick R. Fonseca
University of São Paulo, Brazil
Fondazione Bruno Kessler
rocha@fbk.eu

Simone Magnolini
Fondazione Bruno Kessler
University of Brescia, Italy
magnolini@fbk.eu

Anna Feltracco
Fondazione Bruno Kessler
University of Pavia, Italy
University of Bergamo, Italy
feltracco@fbk.eu

Mohammed R. H. Qwaider
Fondazione Bruno Kessler
Povo-Trento, Italy
qwaider@fbk.eu

Bernardo Magnini
Fondazione Bruno Kessler
Povo-Trento, Italy
magnini@fbk.eu

Abstract

English. We present the system developed at FBK for the EVALITA 2016 Shared Task "QA4FAQ – Question Answering for Frequently Asked Questions". A peculiar characteristic of this task is the total absence of training data, so we created a meaningful representation of the data using only word embeddings. We present the system as well as the results of the two submitted runs, and a qualitative analysis of them.

Italiano. *Presentiamo il sistema sviluppato presso FBK per la risoluzione del task EVALITA 2016 "QA4FAQ - Question Answering for Frequently Asked Questions". Una caratteristica peculiare di questo task é la totale mancanza di dati di training, pertanto abbiamo creato una rappresentazione significativa dei dati utilizzando solamente word embeddings. Presentiamo il sistema assieme ai risultati ottenuti dalle due esecuzioni che abbiamo inviato e un'analisi qualitativa dei risultati stessi.*

1 Introduction

FAQ ranking is an important task inside the wider task of question answering, which represents at the moment a topic of great interest for research and business as well. Analyzing the Frequent Asked Questions is a way to maximize the value of this type of knowledge source that otherwise could be difficult to consult. A similar task was proposed in two SemEval editions (Màrquez et al., 2015) and (Nakov et al., 2016).

Given a knowledge base composed of about 470 questions (henceforth, FAQ question), their respective answers (henceforth, FAQ answers) and metadata (tags), the task consists in retrieving the most relevant FAQ question/answer pair related to the set of queries provided by the organizers.

For this task, no training data were provided, ruling out machine learning based approaches. We took advantage of the *a priori* knowledge provided by word embeddings, and developed a word weighting scheme to produce vector representations of the knowledge base questions, answers and the user queries. We then rank the FAQs with respect to their cosine similarity to the queries.

The paper is organized as follows. Section 2 presents the system we built and Section 3 reports development data we created in order to test our system. In Section 4 we show the results we obtained, followed by Section 5 that presents an error analysis. Finally, Section 6 provides some conclusions.

2 System Description

Our system was based on creating vector representations for each user query (from the test set), question and answer (from the knowledge base), and then ranking the latter two according to the cosine distance to the query.

We created the vectors using the word embeddings generated by Dinu and Baroni (2014) and combined them in a way to give more weight to more important words, as explained below. Since no training data was available, using word embeddings was especially interesting, as they could provide our system with some kind of *a priori* knowl-

edge about similar words.

We applied similar the same operations to queries, FAQ questions and answers, and here we will use the term *text* to refer to any of the three. In order to create vector representations for texts, the following steps were taken:

1. **Tokenization.** The text is tokenized with NLTK's (Bird et al., 2009) Italian model, yielding a token list X.

2. **Filtering.** Stopwords (obtained from NLTK's stopword list) and punctuation signs are discarded from X.

3. **Acronyms Substitution.** Some words and expressions are replaced by their acronyms. We performed this replacement in order to circumvent cases where a query could have an acronym while the corresponding FAQ has the expression fully written, which would lead to a similarity score lower than expected. For example, we replaced *Autorità Idrica Pugliese* with AIP and *Bari* with BA. In total, 21 expressions were checked.

4. **Out-of-vocabolary terms.** When words out of the embedding vocabulary are found in a FAQ question or answer, a random embedding is generated for it[1], from a normal distribution with mean 0 and standard deviation 0.1. The same embedding is used for any new occurrences of that word. This includes any acronyms used in the previous step.

5. **IDF computation.** We compute the document frequency (DF) of each word as the proportion of questions or answers in which it appears[2]. Then, we compute the inverse document frequency (IDF) of words as:

$$\mathrm{IDF}(w) = \begin{cases} \frac{1}{\mathrm{DF}(w)}, & \text{if } \mathrm{DF}(w) > 0 \\ 10, & \text{otherwise} \end{cases} \quad (1)$$

   We found that tweaking the DF by decreasing FAQ tags count could improve our system's performance. When counting words in questions and answers to compute their DF, we ignore any word present among the tags for that FAQ entry. Thus, tag words, which are supposed to be more relevant, have a lower DF and higher IDF value.

6. **Multiword expressions.** We compute the embeddings for 15 common multiword expressions (MWEs) we extracted from the FAQ. They are computed as the average of the embeddings of the MWE components, weighted by their IDF. If an MWE is present in the text, we add a token to X containing the whole expression, but do *not* remove the individual words. An example is *codice cliente*: we add *codice_cliente* to X, but still keep *codice* and *cliente*.

7. **SIDF computation.** We compute the Similarity-IDF (SIDF) scores. This metric can be seen as an extension of the IDF which also incorporates the DF of similar words. It is computed as follows:

$$\mathrm{SIDF}(w) = \frac{1}{\mathrm{SDF}(w)} \quad (2)$$

$$\mathrm{SDF}(w) = \mathrm{DF}(w) + \sum_{w_i \in W_{sim}} \cos(w, w_i)\,\mathrm{DF}(w_i) \quad (3)$$

   Here, W_{sim} denotes the set of the n most similar words to w which have non-zero DF. Note that under this definition, SDF is never null and thus we don't need the special case as in the IDF computation. We can also compute the SIDF for the MWEs introduced to the texts.

8. **Embedding averaging.** After these steps, we take the mean of the embeddings, weighted by the SIDF values of their corresponding words:

$$v = \frac{\sum_{w \in X} \mathrm{E}(w)\,\mathrm{SIDF}(w)}{|X|} \quad (4)$$

   Here, v stands for the vector representation of the text and $\mathrm{E}(\cdot)$ is the function mapping words and MWEs to their embeddings. Note that we do not remove duplicate words.

[1] Out of vocabulary words that only appear in the queries are removed from X.

[2] When we are comparing queries to FAQ questions, we only count occurrences in questions. Likewise, when comparing queries to answers, we only count in answers.

FAQ Entry	id	272
	question	Cos'è la quota fissa riportata in fattura?
	answer	La quota fissa, prevista dal piano tariffario deliberato, è addebitata in ciascuna fattura, fattura, ed è calcolata in base ai moduli contrattuali ed ai giorni di competenza della fattura stessa. La quota fissa è dovuta indipendentemente dal consumo in quanto attiene a parte dei costi fissi che il gestore sostiene per erogare il servizio a tutti. Quindi nella fattura è addebitata proporzionalmente al periodo fatturato.
	tag	fattura, quota, fissa, giorni, canone acqua e fogna, quota fissa, costi fissi, quote fisse
DevSet	paraphrased query	Cosa si intende per quota fissa nella fattura?
	answer-driven query	La quota fissa è indipendente dai consumi?

Table 1: Example of our development set.

In this process, the IDF and SIDF values are calculated independently for answers and questions in the FAQ When processing queries, the value actually used depends on which one we are comparing the query vectors with.

After computing vectors for all texts, we compute the cosine similarity between query vectors and FAQ questions and also between queries and answers. For each FAQ entry, we take the highest value between these two as the system confidence for returning that entry as an answer to the query.

3 Evaluating our system

In order to evaluate our system, we created a development set and we calculate a baseline as a reference threshold.

3.1 Development Set

We manually created a dataset of 293 queries to test our systems. Each query in the dataset is associated to one of the entries provided in the knowledge base. In particular, the dataset is composed by 160 *paraphrased queries* and 133 *answer driven queries*. The *paraphrased queries* are queries obtained by paraphrasing original questions; the *answer queries* are generated without considering the original FAQ questions, but have an answer in the knowledge base. Table 1 shows an example of a *paraphrased query* and an *answer driven query* for FAQ 272 of the knowledge base.

Given the technical domain of the task, most of the generated *paraphrases* recall lexical items of the original FAQ question (e.g. "uso commerciale", "scuola pubblica", etc..). Differently, the *answer driven queries* are not necessarily similar in content and lexicon to the FAQ question; instead we expected it to have a very high similarity with the answer.

We guided the development of our system evaluating it with different versions of this dataset. In particular, version 1 is composed by 200 queries, begin 160 *paraphrased* and 40% answer driven, and version 2 is composed by 266 queries, 133 *paraphrased* and 133 *answer driven*.

Merging *paraphrased queries* and *answer driven queries* (in different proportions) allows us to create a very heterogeneous dataset; we expected the test set and, in general, the questions by users to be as much varied.

3.2 Baseline

Two baseline systems were built using Apache Lucene[3]. *FBK-Baseline-sys1* was built by indexing for each FAQ entry a Document with two fields (id, FAQ question), while *FBK-Baseline-sys2* was built by indexing for each FAQ entry a Document with three fields (id, FAQ question, FAQ answer).

4 Results

In Table 2 we report the results of the two runs of our system compared with the official baseline provided by the organizers. The only difference in our first two runs was that the first one always tried to retrieve an answer, while the second one would abstain from answering when the system confidence was below 0.5.

The organizers baseline (*qa4faq-baseline*[4]) was built using Lucene by having a weighted-index. For each FAQ entry a Document with four fields (id, FAQ question(*weight=4*), FAQ answer(*weight=2*), tag(*weight=1*)).

We use three different metrics to evaluate the system: Accuracy@1, that is the official score to

[3] https://lucene.apache.org/

[4] https://github.com/swapUniba/qa4faq

	Test set		
	Accuracy@1	MAP	Top 10
run 1	35.87	51.12	73.94
run 2	37.46	50.10	71.91
qa4faq-baseline	40.76	58.97	81.71
FBK-Baseline-sys1	39.79	55.36	76.15
FBK-Baseline-sys2	35.16	53.02	80.92

Table 2: Results on the test set. Accuracy@1: official score, MAP: Mean Average Precision, Top 10: correct answer in the first 10 results.

rank the systems, MAP and $Top10$. Accuracy@1 is the precision of the system taking into account only the first answer; it is computed as follows:

$$Accuracy@1 = \frac{(n_c + n_u * \frac{n_c}{n})}{n} \quad (5)$$

Where n_r is the number of correct quries, n_u is the number of unanswered queries and n is the number of questions. MAP is the Mean Average Precision that is the mean of the average precision scores for each query, i.e. the inverse of the ranking of the correct answer. $Top10$ is the percentage of query with the correct answer in the first 10 positions. Both our approach runs underperformed compared with the baseline in all the three metrics we use to evaluate the systems.

Comparing our runs, it is interesting to notice that *run 2* performs better while evaluated with $Accuracy@1$, but worse in the other two metrics; this suggests that, even in some cases where the system confidence was below the threshold, the correct answer was among the top 10.

5 Error Analysis

The results of our system on the development set, described in Section 3.1, compared with the official baseline are reported in Table 3.

As can be seen, both the runs outperform the baseline in every metric, especially in the $Accuracy@1$.

This difference of behavior enlightens that there is a significant difference between the development set and the test set. The systems were developed without knowing the target style, and without training data, so is not surprising that the system is not capable of style adaptation.

An interesting aspect that describes the difference between development set and test set is reported in Table 4: the average and the standard deviation of the number of tokens of every query. In the first line is possible to notice that, not only, our development queries has , in average, more tokens than the test queries, but also that the standard deviation is significantly lower. This distribution of tokens is in line with a qualitative check of the test set. The test set includes incomplete sentences, with only keywords, e.g. *"costo depurazione"*, alongside long questions that include verbose description of the situation e.g. *"Mia figlia acquisterà casa a bari il giorno 22 prossimo. Come procedere per l intestazione dell utenza? Quali documenti occorrono e quali i tempi tecnici necessari?"*. Instead the development set is composed by queries more similar in their structure and well formed.

All systems perform, almost, in the same way according to the data sets: in the two versions of the development set the correct queries are longer with a higher standard deviation compared to the wrong ones; on the other hand, in the test set the correct queries are shorter with a lower standard deviation.

We did a qualitative analysis of the result of our systems; we limited our observation to the 250 queries of the test set for which the right answer was not in the first ten retrieved by our systems. We considered these cases to be the worst and wanted to investigate whether they present an issue that cannot be solved using our approach.

We present in this section some of these cases. In Example 1, the answer of the system is weakly related with the query: the query is very short and its meaning is contained in both the gold standard and in the system answer. In the gold standard the substitution of the counter (*"sostituzione del contatore"*) is the main focus of the sentence, and the other part is just a specification of some detail (*"con saracinesca bloccata"*).

In the system answer the substitution of the counter (*"sostituzione del contatore"*) is the effect of the main focus (*"Per la telelettura"*), but our approach cannot differentiate these two types of text not directly related with the query.

Example 1
Query: *sostituzione del contatore*
Gold standard: *Come effettuare il cambio del contatore vecchio con saracinesca bloccata?*
System answer: *Per la telelettura il contatore sara sostituito con un nuovo contatore?*

A similar issue is visible in Example 2. In this

	Version 1			Version 2		
	Accuracy@1	MAP	Top 10	Accuracy@1	MAP	Top 10
Run 1	72.00	79.64	95.00	66.17	74.77	92.48
Run 2	72.45	77.55	92.00	66.36	73.26	90.23
qa4faq-baseline	69.00	76.22	89.50	60.15	70.22	88.72
FBK-baseline-sys1	49.00	58.63	76.50	39.47	49.53	68.05
FBK-baseline-sys2	52.00	62.69	82.50	49.62	62.10	86.09

Table 3: Results on the development sets. Accuracy@1: official score, MAP: Mean Average Precision, Top 10: correct answer in the first 10 result.

		Version 1	Version 2	Test set
Queries		11.42 +- 4.12	11.20 +- 3.95	7.96 +- 7.27
R1	Answered queries	11.42 +- 4.12	11.20 +- 3.95	7.96 +- 7.27
	Right queries	11.63 +- 4.15	11.41 +- 4.06	7.32+- 5.44
	Wrong queries	10.88 +- 4.00	10.78 +- 3.69	8.32 +- 8.09
R2	Answered queries	11.56 +- 4.12	11.30 +- 3.94	8.09 +- 7.41
	Right queries	11.77 +- 4.12	11.52 +- 4.04	7.37 +- 5.47
	Wrong queries	11.02 +- 4.06	10.86 +- 3.71	8.52 +- 8.33
B	Answered queries	11.42 +- 4.12	11.20 +- 3.95	7.97 +- 7.27
	Right queries	11.94 +- 4.34	11.73 +- 4.35	7.54 +- 5.98
	Wrong queries	10.26 +- 3.31	10.40 +- 3.09	8.26 +- 8.02

Table 4: Average and standard deviation of the number of tokens per query. R1: Run1, R2: Run2, B: Organizers Baseline *qa4faq-baseline*.

case, the first part (*"Quali sono i tempi di allaccio di un contatore"*) of the system answer matches, almost exactly, the query, but as in Example 1, the second part (*"in caso di ripristino in quanto l'abitazione aveva già la fornitura?"*), which is not very relevant to the query, was not enough to reduce the overall ranking of this FAQ. We think this issue could be avoided with some more features, but this would require some training data for a machine learning approach, or some knowledge of the domain to craft a rule approach.

Example 2
Query: *quali sono i tempi di attivazione di un contatore ?*
Gold standard: *Quali sono i tempi previsti per ottenere un allacciamento?*
System answer: *Quali sono i tempi di allaccio di un contatore in caso di ripristino in quanto l'abitazione aveva già la fornitura?*

In some cases, like in Example 3, the semantic match (like common or related words in both sentences) is not enough to understand the relationship, or could me misleading. Some knowledge of the world and some cause-effect reasoning is needed to understand that the gold standard is more related to the query than the system answer. Even if the balance (*"conguaglio"*) and time expressions (*"quando"*, *"luglio e agosto e un po di settembre"*) are present in both query and system answer, and not in the gold standard, they are not useful to find the correct answer.

Example 3
Query: *ho ricevuto una bolletta di conguaglio di e 426.69 , ma son mancata da casa a luglio e agosto e un po di settembre , senza consumare , come mai?*
Gold standard: *Perche ho ricevuto una fattura elevata?*
System answer: *Il conguaglio quando avviene?*

Alongside this issue, there are some cases (Example 4) where our system answers correctly, but due to the semi-automatic nature of the gold standard it has been considered wrong.

Example 4
Query: *chi paga la portella del contatore?*
Gold standard: *Come richiedere la sostituzione dello sportello della nicchia contatore?*
System answer: *Chi paga la portella del contatore?*

Example 5 represents one of the cases in which the systems answer has been considered wrong but is more related with the query than the gold standard.

Example 5
Query: *abito in un condominio con 5 famiglie . se alla scadenza di una bolletta uno dei condomini non vuole pagare la sua quota , possono gli altri 4 pagare la loro parte su un altro bollettino postale?*
Gold standard: *Quali sono le modalita di pagamento delle fatture?*
System answer: *Contratto condominiale, di cui uno moroso come comportarsi?*

6 Conclusion

We reported the system we used in the EVALITA 2016 QA4FAQ shared task, as well as the development set we created to evaluate it and an analysis of our results.

We found that while our system performed below the baseline in the official test set, we had superior performance on our in-house development set. This is apparently related to the different style of the two sets: ours has longer queries, which are more homogeneous with respect to size, while the official one has many very short queries and a few very large ones.

It could be argued that the official test set represents a more realistic scenario than the development set we created, since it contains actual user queries, thus diminishing the relevance of our results. However, further analysis showed that in a number of cases, our system returned a more appropriate FAQ question/answer than what was in the gold standard, due to the gold standard semi-automatic nature.

We hypothesize that our system performed better than what seems from the official results; however, due to the size of the test set, it would be prohibitive to check it manually and arrive at a more precise accuracy.

References

Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Georgiana Dinu and Marco Baroni. 2014. Improving zero-shot learning by mitigating the hubness problem. *arXiv preprint arXiv:1412.6568*.

Lluís Màrquez, James Glass, Walid Magdy, Alessandro Moschitti, Preslav Nakov, and Bilal Randeree. 2015. Semeval-2015 task 3: Answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*.

P Nakov, L Marquez, A Moschitti, W Magdy, H Mubarak, AA Freihat, J Glass, and B Randeree. 2016. Semeval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation. San Diego, California. Association for Computational Linguistics*.

ChiLab4It System in the QA4FAQ Competition

Arianna Pipitone, Giuseppe Tirone, Roberto Pirrone
DIID - Dipartimento dell'Innovazione Industriale e Digitale -
Ingegneria Chimica, Gestionale, Informatica, Meccanica
Università degli Studi di Palermo
{arianna.pipitone, giuseppe.tirone, roberto.pirrone}@unipa.it

Abstract

English. ChiLab4It is the Question Answering system (QA) for Frequently Asked Questions (FAQ) developed by the Computer-Human Interaction Laboratory (ChiLab) at the University of Palermo for participating to the QA4FAQ task at EVALITA 2016 competition. The system is the versioning of the QuASIt framework developed by the same authors, which has been customized to address the particular task. This technical report describes the strategies that have been imported from QuASIt for implementing ChiLab4It, the actual system implementation, and the comparative evaluations with the results of the other participant tools, as provided by the organizers of the task. ChiLab4It was the only system whose score resulted to be above the experimental baseline fixed for the task. A discussion about future extensions of the system is also provided.

Italiano. *ChiLab4It è il sistema di Question Answering (QA) usato per rispondere alle Frequently Asked Questions (FAQs), sviluppato dal Laboratorio di Interazione Uomo-Macchina (Chilab) dell'Università degli Studi di Palermo allo scopo di partecipare al task QA4FAQ nella competizione EVALITA 2016. Il sistema è una versione del framework QuASIt, sviluppato dagli stessi autori e che è stato ridefinito per il task in questione. Il report descrive le strategie che hanno consentito di realizzare ChiLab4It a partire da QuASIt, l'effettiva implementazione del sistema e le valutazioni comparative con gli altri team che hanno partecipato al task, così come sono state rese note dagli organizzatori. ChiLab4It è stato l'unico sistema a superare la baseline sperimentale fissata per il task. Nella parte conclusiva del report, verranno altresì discussi i possibili sviluppi futuri del sistema.*

1 Introduction

This technical report presents ChiLab4It (Pipitone et al., 2016a), the QA system for FAQ developed by the ChiLab at the University of Palermo to attend the QA4FAQ task (Caputo et al., 2016) in the EVALITA 2016 competition (Basile et al., 2016). The main objective of such a task is answering to a natural language question posed by the user by retrieving the more relevant FAQs, among those in the set provided by the Acquedotto Pugliese society (AQP) which developed a semantic retrieval engine for FAQs, called *AQP Risponde*[1]. Such an engine is based on a QA system; it opens new challenges about both the Italian language usage and the variability of language expressions by users.
The background strategy of the proposed tool is based on the cognitive model described in (Pipitone et al., 2016b); in this work the authors present QuASIt, an open-domain QA system for the Italian language, that can be used for both multiple choice and essay questions. When a support text is provided for finding the correct answer (as in the case of text comprehension), QuASIt is able to use this text and find the required information.
ChiLab4It is the customized version of QuASIt to the FAQ domain; such a customization was the result of some restrictions applied on the whole

[1] http://aqprisponde.aqp.it/ask.php

functionalities of QuASIt. The intuition was to consider the FAQ as *support text*; the more relevant FAQ will be the one whose text will best fit the user's question, according to a set of matching strategies that keep into account some linguistic properties, such as typology and syntactic correspondences. The good performances obtained in the evaluations demonstrate the high quality of the idea, although the current linguistic resources for the Italian are not exhaustive. This report is organized as follow: in section 2 the QuASIt system is presented, and in section 3 the ChiLab4It system is described as a restriction of QuASIt. In section 4 the results of ChiLab4It are shown according to the evaluation test bed provided by the competition organizers. Finally, future works are discussed in section 5.

2 The QuASIt System

The main characteristic of QuASIt is the underlying *cognitive* architecture, according to which the interpretation and/or production of a natural language sentence requires the execution of some cognitive processes over both a perceptually grounded model of the world (that is an ontology), and a previously acquired linguistic knowledge.
In particular, two kinds of processes have been devised, that are the *conceptualization of meaning* and the *the conceptualization of form*.
The conceptualization of meaning allows to associate a sense to perceived forms, that are the words of the user query. A sense is the set of concepts of the ontology that explains the form; such a process is implemented considering the ontology nodes whose labels match best the forms from a syntactic point of view. The set of such nodes is the candidate sub-ontology to contain the answer to produce. The syntactic match is based on a syntactic measure.
The second process associates a syntactic expression to a meaning; it implements the strategies for producing the correct form of an answer, once it has been inferred. The form depends on the way QuASIt can be used, that is in both multiple choice and essay questions. In the case of multiple choice questions, the form must be one of the proposed answers. The system infers the correct answer among the proposed ones using the values of the properties' ranges in the sub-ontology; the answer that better syntactically match such ranges is considered the correct answer. If no answer can be inferred in this way, a support text can be used if available. The support text can be either derived automatically by the system, using the plain text associated to the nodes of the sub-ontology (such as an abstract node in the DBPedia ontology[2]) or provided directly to the questions as in the case of a text comprehension task. In figure 1 the architecture of QuASIt is shown. The ontology and the linguistic knowledge are located respectively in the *Domain Ontology Base* and the *Linguistic Base*. The *Mapping to Meanings* (MtM) and the *Mapping to Forms* (MtF) modules are the components that model the cognitive processes related to the conceptualization of meaning and form respectively. The *Unification Merging* module is essentially the FCG engine (Steels, 2011) that is used to perform query parsing.
The strategy we implemented in ChiLab4It system is based on the QuASIt function that selects the correct answer to multiple choice questions using support text; the intuition was that *a FAQ can be considered a support text* that can be used for retrieving the more relevant FAQ to a user's query. For this reason, in the next subsection, we describe this strategy in detail, and next we show how it was applied in the proposed tool.

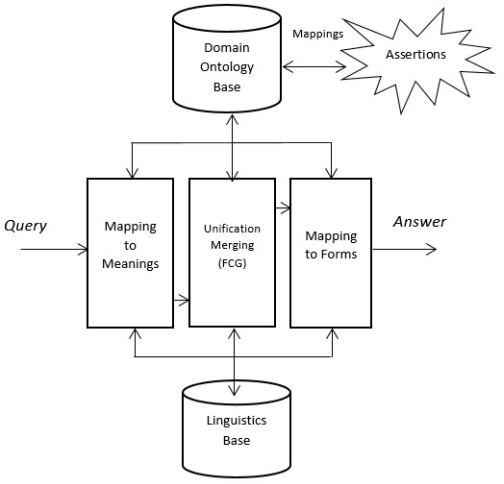


Figure 1: The QuASIt Cognitive Architecture

2.1 Searching in the support text

Searching in a support text is a possible strategy to deal with unstructured information when an artificial agent is trying to answer a particular question. In this case the agent learns a possible answer by comprehending the text dealing with the question

[2]http://it.dbpedia.org/

topic. Such a process is implemented in QuASIt by the MtF module.
Formally, let $Q = \{q_1, q_2 ... q_n\}$ be the query of the user, and $P = \{p_1, p_2, ... p_m\}$ a sentence in the support text; each element in these sets is a token. P will be considered as much similar as Q when maximizing the following similarity measure m:

$$m = |\Im| - (\alpha l + \beta u)$$

where $\Im = \{p_j | \ \exists q_i \in Q, \ J(p_j, q_i) > \tau\}$, and $J(p_j, q_i)$ is the Jaro-Winkler distance between a couple of tokens (Winkler, 1990). As a consequence, $\Im \supset \ Q \cap P$, and $|\Im|$ is the number of matching tokens both in Q and P.
$l = 1 - \frac{|\Im|}{|P|}$ is the number of "lacking tokens" that are tokens belonging to Q that do not match in P, while $u = 1 - \frac{o(Q,\Im)}{|\Im|}$ is the number of "unordered tokens" that is the number of tokens in Q that do not have the same order in $\Im$; here $o(a, b)$ is the function returning maximum number of ordered tokens in a with respect to b.
Both l and u are normalized in the range $[0 \dots 1]$; they are penalty values representing syntactical differences among the sentences. The higher u and l are, the lower is the sentences similarity.
The α and β parameters weight the penalty, and they have been evaluated empirically through experimentation along with τ.
We re-used such strategy in ChiLab4It using different values for α and β parameters depending on which kind of support text we consider during the search, as next explained.

3 ChiLab4It

The basic idea of the proposed tool was to consider a FAQ as a support text. According to the provided dataset, a FAQ is composed by three textual fields: the *question text*, the *answer text* and the *tag set*. For each of these fields we applied the search strategy defined above; in particular we set different α and β parameters for each field in the m measure, depending on linguistics considerations. For this reason, we defined three different parameterized m measures named m_1, m_2 and m_3. Moreover, further improvements were achieved by searching for the synonyms of the words of the query in the answer text. These synonyms were not considered in the QuASIt implementation.
Given the previously defined variables $\Im$, l and u, the α and β parameters were set according to the following considerations:

- *question text*; the α and β parameters are the same of QuASIt, that is $\alpha = 0.1$ and $\beta = 0.2$. This choice is based solely on linguistic motivations; in fact, considering that the support text is a question such as the user query, both sentences to be matched will have interrogative form. As a consequence, both l and u influence the final match. The final measure is:

$$m_1 = |\Im| - (0.1 * l + 0.2 * u)$$

- *answer text*; the search is iterated for each sentence in the text. In this case, the α and β parameters are zero ($\alpha = 0$ and $\beta = 0$). This is because the answer text has a direct form, so the order of tokens must not be considered; moreover, a sentence in the answer text owns more tokens than the query, so this information is not discriminative for the final match.
  In this case, *the search is extended to the synonyms of the words in the query* except to the synonyms of the stop-words; this extension has improved significantly the performances of the system. Empirical evaluations demonstrated that there were not the same improvements when the synonyms were considered for the other parts of a FAQ (question text and tag set) because in these cases the synonyms increase uselessly the number of irrelevant FAQs retrieved by the system.
  Formally, let Σ be the *σ-expansion* set (Pipitone et al., 2014) that contains both the words and the synonyms of such words in the $Q - S_w$ set, being Q the user query as previously defined and S_w the set of stop-words:

$$\Sigma = \{\sigma_i \mid \sigma_i = synset(q_i) \ \wedge \ q_i \in Q - S_w\}$$

  Let's define $S = \{S_1, S_2, \dots, S_N\}$ the set of sentences in the answer text. We defined the M set that contains the m_{s_i} measures computed with $\alpha = 0$ and $\beta = 0$ in m, for each sentence $S_i \in S$ with the σ-expanded query:

$$M = \{m_{s_i} \mid m_{s_i} = |\Im_i|\}$$

  where

$$\Im_i = \{p_j \in S_i \cap \Sigma \mid \exists q_k \in Q, \ J(p_j, q_k) > \tau\}$$

  The final similarity measure m_2 will be the maximum value in M:

$$m_2 = max \ \{m_{s_i} \mid m_{s_i} = |\Im_i|\}$$

- *tag set*; the α and β parameters are zero ($\alpha = 0$ and $\beta = 0$) also in this case. This is because the tags in the set do not own a particular linguistic typology, so the information related to both the order of tokens and the lacking ones must not to be considered. As already explained, the synonyms are not included in this search. As consequence:

$$m_3 = |\Im|$$

where $\Im$ is the previously defined intersection among the query of the user and the set of tags.

A query will be considered as much similar as a FAQ when maximizing the sum of the measures defined previously, so the final similarity value is:

$$m_{faq} = m_1 + m_2 + m_3$$

These values were ordered, and the first 25 FAQs were outputted for a single query as required by the task.

3.1 The architecture

In figure 2 the architecture of ChiLab4It is shown; the input is the query of the user, while the output is the list of the first 25 relevant FAQs. The sources became the *FAQ base* and the *Wiktionary* source from which the provided FAQ dataset and the synonyms are respectively queried.
The white module of such an architecture is the MtF module as implemented in QuASIt. The dark modules are the integrations that have been applied to the MtF module for customizing it to the FAQ domain; in particular, such integrations regard both the σ-expansion of the query and the setting of the analytic form (including parameters) of the m measure depending on the FAQ field.
The first integration is implemented by the *σ module*, that returns the Σ set for the query of the user retrieving the synset from Wiktionary[3].
Parameters and the measure settings are performed by the *FAQ Ctrl* module which is encapsulated into the main MtF module; it retrieves the FAQ from the *FAQ base* and customizes the m measure according to the analyzed field (m_1 for the question text, m_2 for the answer text, m_3 for the tag set). The MtF module computes such measures referring to the σ-expanded query, and finally the m_{faq} value is computed and memorized by the

[3]https://it.wiktionary.org/

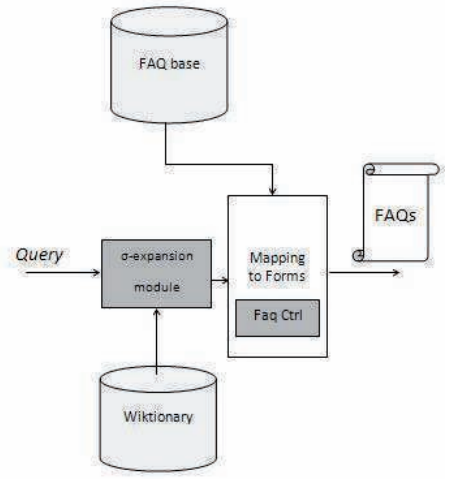


Figure 2: The ChiLab4It Architecture

FAQ Ctrl for tracing the id of the FAQ with the highest value.

3.2 A toy example

In this section we show a toy example with the aim of explaining better the searching process in the support text and how the similarity measure works. Such an example is a real question as retrieved in the data set provided by the organizers.
Let consider the query with $id = 4$, that is: "*a quali orari posso chiamare il numero verde*".
In this case, the Q and the S_w set are:

$$Q = \{A,\ quali,\ orari,\ posso,\ chiamare,\ il,\ numero,\ verde\}$$

and

$$S_w = \{A,\ il\}$$

being "*a*" and "*il*" the stop-words in the question.
The highest measure is computed by ChiLab4It in correspondence to the FAQ with $id = 339$, that is shown in table 1. Considering this FAQ, let compute the three measures for the *question text*, the *answer text* and the *tag set*.
In the first case the support text is the question text of the FAQ, and the P set is:
$P = \{Quali,\ sono,\ gli,\ orari,\ del,\ numero,\ verde\}$ with $|P| = 7$. The m_1 value will be computed considering that the intersection $\Im$ between the question text and the query of the user is:

$$\Im = \{quali,\ orari,\ numero,\ verde\}$$

. The Jaro-Winkler distance is 1 for each word, and $|\Im| = 4$. Also, $l = 1 - \frac{|\Im|}{|P|} = 1 - \frac{4}{7} = 0.428$.

Table 1: The XML description of FAQ 339 as provided in the data set

⟨**faq**⟩

⟨**id**⟩339⟨/**id**⟩

⟨**question**⟩Quali sono gli orari del numero verde?⟨/**question**⟩

⟨**answer**⟩Il servizio del numero verde assistenza clienti AQP 800.085.853 e attivo dal lunedi al venerdi dalle ore 08.30 alle 17.30, il sabato dalle 08.30 alle 13.00; il servizio del numero verde segnalazioni guasto 800.735.735 e attivo 24 ore su 24.⟨/**answer**⟩

⟨**tag**⟩informazioni, orari, numero verde⟨/**tag**⟩

⟨/**faq**⟩

For the calculation of u, we notice that $o(Q, \Im)$ returns 4 because the tokens in Q are all ordered with respect to $\Im$, that means they follow the same sequence in $\Im$. As consequence, $u = 1 - \frac{o(Q,\Im)}{|\Im|} = 1 - \frac{4}{4} = 0$. Substituting all values, m_1 will be:

$$m_1 = |\Im| - (0.1 * l + 0.2 * u) = 3.95$$

In the next step, we consider the answer text; in the FAQ, this text is composed by only one sentence that becomes the new support text P, and the procedure will be applied once. In particular, $S = \{S_1\}$ and $P = S_1 = \{$*Il, servizio, del, numero, verde, assistenza, clienti,...., attivo, 24, ore, su, 24*$\}$ as shown in table 1. In this case, the m_2 measure depends only from the intersection between the σ-expanded query and S_1. In particular, the Σ set is computed unifying the difference set $Q - S_w = \{Quali, orari, posso, chiamare, numero, verde\}$ with the synset from Wiktionary of each such token, so: Σ = {*[[quali], [orari], [posso], [chiamare, soprannominare, chiedere, richiedere], [numero, cifra, contrassegno numerico, matricola, buffone, pagliaccio, elenco, gruppo, serie, classe, gamma, schiera, novero, taglia, misura, attrazione, scenetta, sketch, esibizione, gag, sagoma, macchietta, fascicolo, puntata, dispensa, copia, tagliando, contrassegno, talloncino, titoli, dote, requisito], [verde, pallido, smorto, esangue, acerbo, giovanile, vivace, vigoroso, florido, verdeggiante, lussureggiante, rigoglioso, agricolo, agrario, vegetazione, vigore, rigoglio, freschezza, floridezza, via, avanti, ecologista, ambientalista, livido]]*}, where the synsets are represented in square brackets for clarity. The intersection $\Im_1 = \Sigma \cap S_1$ is simple $\Im_1 = \{numero, verde, orari\}$ because these tokens have the highest Jaro-Winkler distance from the tokens in S_1. As consequence, $M = \{|\Im_1|\} = \{3\}$ and $m_2 = 3$.
In the third case, the support text is the tag set, so $P = \{informazioni,\ orari,\ numero,\ verde\}$ and $\Im = \{orari,\ numero,\ verde\}$. The m_3 value is simply $m_3 = |\Im| = 3$.
Finally, the m measure is computed adding the three calculated values, so $m = 3.95 + 3 + 3 = 9.95$ that represents the highest value among those computed for all FAQs in the dataset.

4 Evaluations

The dataset used for the evaluation was the one provided by the QA4FAQ task organizers; they released such a dataset as a collection of both questions and feedbacks that real customers provided to the AQP Risponde engine.
In particular, such dataset includes:

- a knowledge base of about 470 FAQs, each composed by the text fields we referred to;
- a set of query by customers;
- a set of pairs that allows organizers to evaluate the possible contestants. The organizers analyzed the feedbacks provided by real customers of AQP Risponde engine, and checked them for removing noise.

Training data were not provided: in fact AQP is interested in the development of unsupervised systems, like ChiLab4It is.
According to the guideline, we provided results in a text file purposely formatted, and for each query in the dataset we considered the first 25 answers. However, only the first FAQ is considered relevant for the scope of the task. ChiLab4It is ranked according to the *accuracy@1* (c@1), whose formulation is:

$$c@1 = \frac{1}{n}(n_R + n_U \frac{n_R}{n})$$

where n_R is the number of correct answers, n_U is the number of unanswered questions, and n is the total number of questions.
A participant could have provided two different runs, but in our case we considered only the best configuration of the system. In table 2 we show

Table 2: The final results for QA4FAQ task

TEAM	c@1
ChiLab4It	**0.4439**
baseline	*0.4076*
Team 1 run 1	0.3746
Team 1 run 2	0.3587
Team 2 run 1	0.2125
Team 2 run 2	0.0168

the final results with the ranks of all participants as provided by the organizers; our tool performed better than the other participants, and it was the only one ranked above the experimental baseline.

5 Discussion and Future Works

ChiLab4It has been presented in this work, that is a tool designed for participating to the QA4FAQ task in the EVALITA 2016 competition. ChiLab4It relies on QuASIt, a cognitive model for an artificial agent performing question answering in Italian, already presented by the authors. QuASIt is able to answer both multiple choice and essay questions using an ontology-based approach where the agents manages both domain and linguistic knowledge.

ChiLab4It uses the functions of QuASIt aimed at answering multiple choice questions using a support text to understand the query because a FAQ can be regarded exactly as a support text, that can be used to understand the query sentence and to provide the answer. Moreover our tool enhances the sentence similarity measure introduced in our reference cognitive model in two ways. First, three separate measures are computed for the three parts of a FAQ that is question text, answer text and tag set, and they are summed to provide the final similarity. Second, the synonyms of the query words are analyzed to match the query against each sentence of the answer text of the FAQ to achieve linguistic flexibility when searching for the query topic inside each text.

ChiLab4It was tested with the competition data, and it resulted to be the winner having a c@1 rank well above the fixed experimental baseline.

Future works are aimed at refining the development of the entire QuASIT system. Particular attention will be devoted in studying more refined versions of the similarity measure to take into account complex phrasal structures.

References

Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli. 2016. EVALITA 2016: Overview of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. aAcademia University Press.

Annalina Caputo, Marco de Gemmis, Pasquale Lops, Franco Lovecchio, and Vito Manzari. 2016. Overview of the EVALITA 2016 Question Answering for Frequently Asked Questions (QA4FAQ) Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Arianna Pipitone, Vincenzo Cannella, and Roberto Pirrone. 2014. I-ChatbIT: an Intelligent Chatbot for the Italian Language. In Roberto Basile, Alessandro Lenci, and Bernardo Magnini, editors, *Proceedings of the First Italian Conference on Computational Linguistics CLiC-it 2014*. Pisa University Press.

Arianna Pipitone, Giuseppe Tirone, and Roberto Pirrone. 2016a. Chilab4IT: ChiLab4It System in the QA4FAQ Competition. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Arianna Pipitone, Giuseppe Tirone, and Roberto Pirrone. 2016b. QuASIt: a Cognitive Inspired Approach to Question Answering System for the Italian Language. In *Proceedings of the 15th International Conference on the Italian Association for Artificial Intelligence 2016*. in press.

Luc Steels, 2011. *Introducing Fluid Construction Grammar*. John Benjamins.

William E. Winkler. 1990. String comparator metrics and enhanced decisionn rules in the fellegi-sunter model of record linkage. In *Proceedings of the Section on Survey Research*, pages 354–359.

Overview of the Evalita 2016 SENTIment POLarity Classification Task

Francesco Barbieri
Pompeu Fabra University
Spain
francesco.barbieri@upf.edu

Valerio Basile
Université Côte d'Azur, Inria, CNRS, I3S
France
valerio.basile@inria.fr

Danilo Croce
University of Rome "Tor Vergata"
Italy
croce@info.uniroma2.it

Malvina Nissim
University of Groningen
The Netherlands
m.nissim@rug.nl

Nicole Novielli
University of Bari "A. Moro"
Italy
nicole.novielli@uniba.it

Viviana Patti
University of Torino
Italy
patti@di.unito.it

Abstract

English. The SENTIment POLarity Classification Task 2016 (SENTIPOLC), is a rerun of the shared task on sentiment classification at the message level on Italian tweets proposed for the first time in 2014 for the Evalita evaluation campaign. It includes three subtasks: *subjectivity classification*, *polarity classification*, and *irony detection*. In 2016 SENTIPOLC has been again the most participated EVALITA task with a total of 57 submitted runs from 13 different teams. We present the datasets – which includes an enriched annotation scheme for dealing with the impact on polarity of a figurative use of language – the evaluation methodology, and discuss results and participating systems.

Italiano. *Descriviamo modalità e risultati della seconda edizione della campagna di valutazione di sistemi di sentiment analysis (SENTIment POLarity Classification Task), proposta nel contesto di "EVALITA 2016: Evaluation of NLP and Speech Tools for Italian". In SENTIPOLC è stata valutata la capacità dei sistemi di riconoscere diversi aspetti del sentiment espresso nei messaggi Twitter in lingua italiana, con un'articolazione in tre sottotask:* subjectivity classification*,* polarity classification *e* irony detection*. La campagna ha suscitato nuovamente grande interesse, con un totale di* 57 *run inviati da* 13 *gruppi di partecipanti.*

1 Introduction

Sentiment classification on Twitter, namely detecting whether a tweet is polarised towards a positive or negative sentiment, is by now an established task. Such solid and growing interest is reflected in the fact that the Sentiment Analysis tasks at SemEval (where they constitute now a whole track) have attracted the highest number of participants in the last years (Rosenthal et al., 2014; Rosenthal et al., 2015; Nakov et al., 2016), and so it has been for the latest Evalita campaign, where a sentiment classification task (SENTIPOLC 2014) was introduced for the first time (Basile et al., 2014).

In addition to detecting the *polarity* of a tweet, it is also deemed important to detect whether a tweet is *subjective* or is merely reporting some fact, and whether some form of figurative mechanism, chiefly *irony*, is also present. Subjectivity, polarity, and irony detection form the three tasks of the SENTIPOLC 2016 campaign, which is a rerun of SENTIPOLC 2014.

Innovations with respect to SENTIPOLC 2014 While the three tasks are the same as those organised within SENTIPOLC 2014, we want to highlight the innovations that we have included in this year's edition. First, we have introduced two new annotation fields which express *literal polarity*, to provide insights into the mechanisms behind polarity shifts in the presence of figurative usage. Second, the test data is still drawn from Twitter, but it is composed of a portion of random tweets and a portion of tweets selected via keywords, which do not exactly match the selection procedure that led to the creation of the training set. This was intentionally done to observe the portability of supervised systems, in line with what observed in (Basile et al., 2015). Third, a portion of the data was annotated via Crowdflower rather than by experts. This has led to several observations on the quality of the data, and on the theoretical description of the task itself. Fourth, a portion

of the test data overlaps with the test data from three other tasks at Evalita 2016, namely PoST-WITA (Bosco et al., 2016), NEEL-IT (Basile et al., 2016a), and FactA (Minard et al., 2016). This was meant to produce a layered annotated dataset where end-to-end systems that address a variety of tasks can be fully developed and tested.

2 Task description

As in SENTIPOLC 2014, we have three tasks.

Task 1: Subjectivity Classification: *a system must decide whether a given message is subjective or objective* (Bruce and Wiebe, 1999; Pang and Lee, 2008).

Task 2: Polarity Classification: *a system must decide whether a given message is of positive, negative, neutral or mixed sentiment.* Differently from most SA tasks (chiefly the Semeval tasks) and in accordance with (Basile et al., 2014), in our data positive and negative polarities are *not* mutually exclusive and each is annotated as a binary category. A tweet can thus be at the same time positive *and* negative, yielding a mixed polarity, or also neither positive nor negative, meaning it is a subjective statement with neutral polarity.[1] Section 3 provides further explanation and examples.

Task 3: Irony Detection: *a system must decide whether a given message is ironic or not.* Twitter communications include a high percentage of ironic messages (Davidov et al., 2010; Hao and Veale, 2010; González-Ibáñez et al., 2011; Reyes et al., 2013; Reyes and Rosso, 2014), and platforms monitoring the sentiment in Twitter messages experienced the phenomenon of wrong polarity classification in ironic messages (Bosco et al., 2013; Ghosh et al., 2015). Indeed, ironic devices in a text can work as unexpected "polarity reversers" (one says something "good" to mean something "bad"), thus undermining systems' accuracy. In this sense, though not including a specific task on its detection, we have added an annotation layer of *literal polarity* (see Section 3.2) which could be potentially used by systems, and also allows us to observe patterns of irony.

The three tasks are meant to be independent. For example, a team could take part in the polarity classification task without tackling Task 1.

[1] In accordance with (Wiebe et al., 2005).

3 Development and Test Data

Data released for the shared task comes from different datasets. We re-used the whole SENTIPOLC 2014 dataset, and also added new tweets derived from different datasets previously developed for Italian. The dataset composition has been designed in cooperation with other Evalita 2016 tasks, in particular the Named Entity rEcognition and Linking in Italian Tweets shared task (NEEL-IT, Basile et al. (2016a)). The multiple layers of annotation are intended as a first step towards the long-term goal of enabling participants to develop end-to-end systems from entity linking to entity-based sentiment analysis (Basile et al., 2015). A portion of the data overlaps with data from NEEL-IT (Basile et al., 2016a), PoSTWITA (Bosco et al., 2016) and FacTA (Minard et al., 2016). See (Basile et al., 2016b) for details.

3.1 Corpora Description

Both training and test data developed for the 2014 edition of the shared task were included as training data in the 2016 release. Summarizing, the data that we are using for this shared task is a collection of tweets which is partially derived from two existing corpora, namely Sentipolc 2014 (TW-SENTIPOLC14, 6421 tweets) (Basile et al., 2014), and TWitterBuonaScuola (TW-BS) (Stranisci et al., 2016), from which we selected 1500 tweets. Furthermore, two new sets have been annotated from scratch following the SENTIPOLC 2016 annotation scheme: the first one consists of a set of 1500 tweets selected from the TWITA 2015 collection (TW-TWITA15, Basile and Nissim (2013)), the second one consists of 1000 (reduced to 989 after eliminating malformed tweets) tweets collected in the context of the NEEL-IT shared task (TW-NEELIT, Basile et al. (2016a)). The subsets of data extracted from existing corpora (TW-SENTIPOLC14 and TW-BS) have been revised according to the new annotation guidelines specifically devised for this task (see Section 3.3 for details).

Tweets in the datasets are marked with a "topic" tag. The training data includes both a *political* collection of tweets and a *generic* collection of tweets. The former has been extracted exploiting specific keywords and hashtags marking political topics ($topic = 1$ in the dataset), while the latter is composed of random tweets on any topic ($topic = 0$). The test material includes tweets from the

TW-BS corpus, that were extracted with a specific *socio-political* topic (via hashtags and keywords related to #labuonascuola, different from the ones used to collect the training material). To mark the fact that such tweets focus on a different topic they have been marked with $topic = 2$. While SENTIPOLC does not include any task which takes the "topic" information into account, we release it in case participants want to make use of it.

3.2 Annotation Scheme

Six fields contain values related to manual annotation are: `subj`, `opos`, `oneg`, `iro`, `lpos`, `lneg`.

The annotation scheme applied in SENTIPOLC 2014 has been enriched with two new fields, `lpos` and `lneg`, which encode the *literal* positive and negative polarity of tweets, respectively. Even if SENTIPOLC does not include any task which involves the actual classification of literal polarity, this information is provided to enable participants to reason about the possible polarity inversion due to the use of figurative language in ironic tweets. Indeed, in the presence of a figurative reading, the literal polarity of a tweet might differ from the intended overall polarity of the text (expressed by `opos` and `oneg`). Please note the following issues about our annotation scheme:

- An objective tweet will not have any polarity nor irony, thus if `subj` $= 0$, then `opos` $= 0$, `oneg` $= 0$, `iro` $= 0$, `lpos` $= 0$, and `lneg` $= 0$.
- A subjective, non ironic, tweet can exhibit at the same time *overall* positive *and* negative polarity (mixed polarity), thus `opos` $= 1$ and `oneg` $= 1$ can co-exist. Mixed *literal* polarity might also be observed, so that `lpos` $= 1$ and `lneg` $= 1$ can co-exist, and this is true for both non-ironic and ironic tweets.
- A subjective, non ironic, tweet can exhibit no specific polarity and be neutral but with a subjective flavor, thus `subj` $= 1$ and `opos` $= 0$, `oneg` $= 0$. Neutral *literal* polarity might also be observed, so that `lpos` $= 0$ and `lneg` $= 0$ is a possible combination; this is true for both non-ironic and ironic tweets.
- An ironic tweet is always subjective and it must have one defined polarity, so that `iro` $= 1$ cannot be combined with `opos` and `oneg` having the same value. However, mixed or neutral literal polarity could be observed for ironic tweets. Therefore, `iro` $= 1$, `lpos` $= 0$, and `lneg` $= 0$ can co-exist, as well as `iro` $= 1$, `lpos` $= 1$, and `lneg` $= 1$.
- For subjective tweets without irony (`iro` $= 0$), the overall (`opos` and `oneg`) and the literal (`lpos` and `lneg`) polarities are always annotated consistently, i.e. `opos` $=$ `lpos` and `oneg` $=$ `lneg`. Note that in such cases the literal polarity is implied automatically from the overall polarity and not annotated manually. The manual annotation of literal polarity only concerns tweets with `iro` $= 1$.

Table 1 summarises the allowed combinations.

3.3 Annotation procedure

Annotations for data from existing corpora (TW-BS and TW-SENTIPOLC14) have been revised and completed by exploiting an annotation procedure which involved a group of six expert annotators, in order to make them compliant to the SENTIPOLC 2016 annotation scheme. Data from NEEL-IT and TWITA15 was annotated from scratch using CrowdFlower. Both training and test data included a mixture of data annotated by experts and crowd. In particular, the whole TW-SENTIPOLC14 has been included in the development data release, while TW-BS was included in the test data release. Moreover, a set of 500 tweets from crowdsourced data was included in the test set, after a manual check and re-assessment (see below: *Crowdsourced data: consolidation of annotations*). This set contains the 300 tweets used as test data in the PoSTWITA, NEEL-IT-it and FactA EVALITA 2016 shared tasks.

TW-SENTIPOLC14 Data from the previous evaluation campaign didn't include any distinction between literal and overall polarity. Therefore, the old tags `pos` and `neg` were automatically mapped into the new labels `opos` and `oneg`, respectively, which indicate overall polarity. Then, we had to extend the annotation to provide labels for positive and negative literal polarity. In case of tweets without irony, literal polarity values were implied from the overall polarity. For ironic tweets, instead, i.e. `iro` $= 1$ (806 tweets), we resorted to manual annotation: for each tweet, two independent annotations have been provided for the literal polarity dimension. The inter-annotator agreement at this stage was $\kappa = 0.538$. In a second round, a third independent annotation was provided to solve the disagreement. The final label

Table 1: Combinations of values allowed by our annotation scheme

subj	opos	oneg	iro	lpos	lneg	description and *explanatory tweet in Italian*
0	0	0	0	0	0	objective *l'articolo di Roberto Ciccarelli dal manifesto di oggi* `http://fb.me/1BQVy5WAk`
1	0	0	0	0	0	subjective with neutral polarity and no irony *Primo passaggio alla #strabrollo ma secondo me non era un iscritto*
1	1	0	0	1	0	subjective with positive polarity and no irony *splendida foto di Fabrizio, pluri cliccata nei siti internazionali di Photo Natura* `http://t.co/GWoZqbxAuS`
1	0	1	0	0	1	subjective with negative polarity and no irony *Monti, ripensaci: l'inutile Torino-Lione inguaia l'Italia: Tav, appello a Mario Monti da Mercalli, Cicconi, Pont...* `http://t.co/3CazKS7Y`
1	1	1	0	1	1	subjective with both positive and negative polarity (mixed polarity) and no irony *Dati negativi da Confindustria che spera nel nuovo governo Monti. Castiglione: "Avanti con le riforme"* `http://t.co/kIKnbFY7`
1	1	0	1	1	0	subjective with positive polarity, and an ironic twist *Questo governo Monti dei paschi di Siena sta cominciando a carburare; speriamo bene...*
1	1	0	1	0	1	subjective with positive polarity, an ironic twist, and negative literal polarity *Non riesco a trovare nani e ballerine nel governo Monti. Ci deve essere un errore! :)*
1	0	1	1	0	1	subjective with negative polarity, and an ironic twist *Calderoli: Governo Monti? Banda Bassotti ..infatti loro erano quelli della Magliana.. #FullMonti #fuoritutti #piazzapulita*
1	0	1	1	1	0	subjective with negative polarity, an ironic twist, and positive literal polarity *Ho molta fiducia nel nuovo Governo Monti. Più o meno la stessa che ripongo in mia madre che tenta di inviare un'email.*
1	1	0	1	0	0	subjective with positive polarity, an ironic twist, and neutral literal polarity *Il vecchio governo paragonato al governo #monti sembra il cast di un film di lino banfi e Renzo montagnani rispetto ad uno di scorsese*
1	0	1	1	0	0	subjective with negative polarity, an ironic twist, and neutral literal polarity *arriva Mario #Monti: pronti a mettere tutti il grembiulino?*
1	1	0	1	1	1	subjective with positive polarity, an ironic twist, and mixed literal polarity *Non aspettare che il Governo Monti prenda anche i tuoi regali di Natale... Corri da noi, e potrai trovare IDEE REGALO a partire da 10e...*
1	0	1	1	1	1	subjective with negative polarity, an ironic twist, and mixed literal polarity *applauso freddissimo al Senato per Mario Monti. Ottimo.*

was assigned by majority vote on each field independently. With three annotators, this procedure ensures an unambiguous result for every tweet.

TW-BS The TW-BS section of the dataset had been previously annotated for polarity and irony[2]. The original TW-BS annotation scheme, however, did not provide any separate annotation for overall and literal polarity. The tags POS, NEG, MIXED and NONE, HUMPOS, HUMNEG in TW-BS were automatically mapped in the following values for the SENTIPOLC's `subj`, `opos`, `oneg`, `iro`, `lpos` and `lneg` annotation fields: POS $\Rightarrow$ 110010; NEG $\Rightarrow$ 101001; MIXED $\Rightarrow$ 111011; NONE $\Rightarrow$ 000000[3]; HUMPOS $\Rightarrow$ 1101??; HUMNEG $\Rightarrow$ 1011??. For the last two cases, i.e. where `iro=1`, the same manual annotation procedure described above was applied to obtain literal polarity values: two independent annotations were provided (inter-annotator agreement $\kappa = 0.605$), and a third annotation was added in a second round in cases of disagreement. Just as with the TW-SENTIPOLC14 set, the final label assignment was done by majority vote on each field.

TW-TWITA15 and TW-NEEL-IT For these new datasets, all fields were annotated from scratch using CrowdFlower (CF)[4], a crowdsourcing platform which has also been recently used for a similar annotation task (Nakov et al., 2016). CF enables quality control of the annotations across a number of dimensions, also by employing test questions to find and exclude unreliable annotators. We gave the users a series of guidelines in Italian, including a list of examples of tweets and their annotation according to the SENTIPOLC scheme. The guidelines also contained an explanation of the rules we followed for the annotation of the rest of the dataset, although in practice these constraints were not enforced in the CF

[2]For the annotation process and inter-annotator agreement see (Stranisci et al., 2016)

[3]Two independent annotators reconsidered the set of tweets tagged by NONE in order to distinguish the few cases of subjective, neutral, not-ironic tweets, i.e. 100000, as the original TW-BS scheme did not allow such finer distinction. The inter-annotator agreement on this task was measured as $\kappa = 0.841$ and a third independent annotation was used to solve the few cases of disagreement.

[4]`http://www.crowdflower.com/`

interface. As requested by the platform, we provided a restricted set of "correct" answers to test the reliability of the users. This step proved to be challenging, since in many cases the annotation of at least one dimension is not clear cut. We required to collect at least three independent judgments for each tweet. The total cost of the crowdsourcing has been 55 USD and we collected 9517 judgments in total from 65 workers. We adopted the default CF settings for assigning the majority label (relative majority). The CF reported average confidence (i.e., inter-rater agreement) is 0.79 for subjectivity, 0.89 for positive polarity (0.90 for literal positivity), 0.91 for negative polarity (0.93 for literal negativity) and 0.92 for irony. While such scores appear high, they are skewed towards the over-assignment of the "0" label for basically all of classes (see below for further comments on this). Percentage agreement on the assignment of "1" is much lower (ranging from 0.70 to 0.77).[5] On the basis of such observations and on a first analysis of the resulting combinations, we operated a few revisions on the crowd-collected data.

Crowdsourced data: consolidation of annotations Despite having provided the workers with guidelines, we identified a few cases of value combinations that were not allowed in our annotation scheme, e.g., ironic or polarised tweets (positive, negative or mixed) which were not marked as subjective. We automatically fixed the annotation for such cases, in order to release datasets of only tweets annotated with labels consistent with the SENTIPOLC's annotation scheme.[6]

Moreover, we applied a further manual check of crowdsourced data stimulated by the following observations. When comparing the distributions of values (0,1) for each label in both training and crowdsourced test data, we observed, as mentioned above, that while the assignment of 1s constituted from 28 to 40% of all assignments for the `opos`/`pos`/ `oneg`/`neg` labels, and about 68% for the subjectivity label, figures were much lower for the crowdsourced data, with percentages as low as

[5]This would be taken into account if using Kappa, which is however an unsuitable measure in this context due to the varying number of annotators per instance.

[6]In particular, for CF data we applied two automatic transformations for restoring consistency of configurations of annotated values in cases where we observed a violation of the scheme: when at least a value 1 is present in the fields `opos`, `oneg`, `iro`, `lpos`, or `lneg`, we set the field `subj` accordingly: `subj`=0 ⇒ `subj`=1; when `iro`=0, the literal polarity value is overwritten by the overall polarity value.

Table 2: Distribution of value combinations

combination						dev	test
subj	opos	oneg	iro	lpos	lneg		
0	0	0	0	0	0	2,312	695
1	0	0	0	0	0	504	219
1	0	1	0	0	1	1,798	520
1	0	1	1	0	0	210	73
1	0	1	1	0	1	225	53
1	0	1	1	1	0	239	66
1	0	1	1	1	1	71	22
1	1	0	0	1	0	1,488	295
1	1	0	1	0	0	29	3
1	1	0	1	0	1	22	4
1	1	0	1	1	0	62	8
1	1	0	1	1	1	10	6
1	1	1	0	1	1	440	36
total						7,410	2,000

6 (`neg`), 9 (`pos`), 11 (`oneg`), and 17 (`opos`), and under 50% for subj.[7] This could be an indication of a more conservative interpretation of sentiment on the part of the crowd (note that 0 is also the default value), possibly also due to too few examples in the guidelines, and in any case to the intrinsic subjectivity of the task. On such basis, we decided to add two more expert annotations to the crowd-annotated test-set, and take the majority vote from *crowd*, *expert1*, and *expert2*. This does not erase the contribution of the crowd, but hopefully maximises consistency with the guidelines in order to provide a solid evaluation benchmark for this task.

3.4 Format and Distribution

We provided participants we a single development set, which consists of a collection of 7,410 tweets, with IDs and annotations concerning all three SENTIPOLC's subtasks: subjectivity classification (`subj`), polarity classification (`opos`,`oneg`) and irony detection (`iro`).

Including the two additional fields with respect to SENTIPOLC 2014, namely `lpos` and `lneg`, the final data format of the distribution is as follows: "`id`", "`subj`, "`opos`", "`oneg`", "`iro`", "`lpos`", "`lneg`", "`top`", "`text`".

The development data includes for each tweet the manual annotation for the `subj`, `opos`, `oneg`, `iro`, `lpos` and `lneg` fields, according to the format explained above. Instead, the blind version of the test data, which consists of 2000 tweets, only contains values for the `idtwitter` and `text` fields. In other words, the development data contains the six columns manually annotated,

[7]The annotation of the presence of irony shows less distance, with 12% in the training set and 8% in the crowd-annotated test set.

while the test data will contain values only in the first (`idtwitter`) and last two columns (`top` and `text`). The literal polarity might be predicted and used by participants to provide the final classification of the items in the test set, however this should be specified in the submission phase. The distribution of combinations in both development and test data is given in Table 2.

4 Evaluation

Task1: subjectivity classification. Systems are evaluated on the assignment of a 0 or 1 value to the subjectivity field. A response is considered plainly correct or wrong when compared to the gold standard annotation. We compute precision (p), recall (r) and F-score (F) for each class (`subj,obj`):

$$p_{class} = \frac{\#correct_{class}}{\#assigned_{class}} \quad r_{class} = \frac{\#correct_{class}}{\#total_{class}}$$
$$F_{class} = 2\frac{p_{class}\, r_{class}}{p_{class} + r_{class}}$$

The overall F-score will be the average of the F-scores for subjective and objective classes.

Task2: polarity classification. Our coding system allows for four combinations of `opos` and `oneg` values: 10 (positive polarity), 01 (negative polarity), 11 (mixed polarity), 00 (no polarity). Accordingly, we evaluate positive and negative polarity independently by computing precision, recall and F-score for both classes (0 and 1):

$$p^{pos}_{class} = \frac{\#correct^{pos}_{class}}{\#assigned^{pos}_{class}} \quad r^{pos}_{class} = \frac{\#correct^{pos}_{class}}{\#total^{pos}_{class}}$$
$$p^{neg}_{class} = \frac{\#correct^{neg}_{class}}{\#assigned^{neg}_{class}} \quad r^{neg}_{class} = \frac{\#correct^{neg}_{class}}{\#total^{neg}_{class}}$$
$$F^{pos}_{class} = 2\frac{p^{pos}_{class}\, r^{pos}_{class}}{p^{pos}_{class} + r^{pos}_{class}} \quad F^{neg}_{class} = 2\frac{p^{neg}_{class}\, r^{neg}_{class}}{p^{neg}_{class} + r^{neg}_{class}}$$

The F-score for the two polarity classes is the average of the F-scores of the respective pairs:

$$F^{pos} = \frac{(F^{pos}_0 + F^{pos}_1)}{2} \quad F^{neg} = \frac{(F^{neg}_0 + F^{neg}_1)}{2}$$

Finally, the overall F-score for Task 2 is given by the average of the F-scores of the two polarities.

Task3: irony detection. Systems are evaluated on their assignment of a 0 or 1 value to the irony field. A response is considered fully correct or wrong when compared to the gold standard annotation. We measure precision, recall and F-score for each class (`ironic,non-ironic`), similarly to the Task1, but with different targeted classes. The overall F-score will be the average of the F-scores for ironic and non-ironic classes.

Informal evaluation of literal polarity classification. Our coding system allows for four combinations of `positive` (`lpos`) and `negative` (`lneg`) values for literal polarity, namely: 10: positive literal polarity; 01: negative literal polarity; 11: mixed literal polarity; 00: no polarity.

SENTIPOLC does not include any task that explicitly takes into account the evaluation of literal polarity classification. However, participants could find it useful in developing their system, and might learn to predict it. Therefore, they could choose to submit also this information to receive an informal evaluation of the performance on these two fields, following the same evaluation criteria adopted for Task 2. The performance on the literal polarity classification will not affect in any way the final ranks for the three SENTIPOLC tasks.

5 Participants and Results

A total of 13 teams from 6 different countries participated in at least one of the three tasks of SENTIPOLC. Table 3 provides an overview of the teams, their affiliation, their country (C) and the tasks they took part in.

Table 3: Teams participating to SENTIPOLC 2016

team	institution	C	tasks
ADAPT	Adapt Centre	IE	T1,T2,T3
CoLingLab	CoLingLab University of Pisa	IT	T2
CoMoDI	FICLIT University of Bologna	IT	T3
INGEOTEC	CentroGEO/INFOTEC CONACyT	MX	T1,T2
IntIntUniba	University of Bari	IT	T2
IRADABE	Univer. Pol. de Valencia, Université de Paris	ES,FR	T1,T2,T3
ItaliaNLP	ItaliaNLP Lab ILC (CNR)	IT	T1,T2,T3
samskara	LARI Lab, ILC CNR	IT	T1,T2
SwissCheese	Zurich University of Applied Sciences	CH	T1,T2,T3
tweet2check	Finsa s.p.a.	IT	T1,T2,T3
UniBO	University of Bologna	IT	T1,T2
UniPI	University of Pisa	IT	T1,T2
Unitor	University of Roma Tor Vergata	IT	T1,T2,T3

Almost all teams participated to both subjectivity and polarity classification subtasks. Each team had to submit at least a constrained run. Furthermore, teams were allowed to submit up to four runs (2 constrained and 2 unconstrained) in

case they implemented different systems. Overall we have 19, 26, 12 submitted runs for the subjectivity, polarity, and irony detection tasks, respectively. In particular, three teams (UniPI, Unitor and tweet2check) participated with both a constrained and an unconstrained runs on the both the subjectivity and polarity subtasks. Unconstrained runs were submitted to the polarity subtask only by IntIntUniba.SentiPy and INGEOTEC.B4MSA. Differently from SENTIPOLC 2014, unconstrained systems performed better than constrained ones, with the only exception of UniPI, whose constrained system ranked first for the polarity classification subtask.

We produced a single-ranking table for each subtask, where unconstrained runs are properly marked. Notice that we only use the final F-score for global scoring and ranking. However, systems that are ranked midway might have excelled in precision for a given class or scored very bad in recall for another.[8]

For each task, we ran a majority class baseline to set a lower-bound for performance. In the tables it is always reported as *Baseline*.

5.1 Task1: subjectivity classification

Table 4 shows results for the subjectivity classification task, which attracted 19 total submissions from 10 different teams. The highest F-score is achieved by `Unitor` at 0.7444, which is also the best unconstrained performance. Among the constrained systems, the best F-score is achieved by samskara with $F = 0.7184$. All participating systems show an improvement over the baseline.

5.2 Task2: polarity classification

Table 5 shows results for polarity classification, the most popular subtask with 26 submissions from 12 teams. The highest F-score is achieved by UniPi at 0.6638, which is also the best score among the constrained runs. As for unconstrained runs, the best performance is achieved by Unitor with $F = 0.6620$. All participating systems show an improvement over the baseline.[9]

[8] Detailed scores for all classes and tasks are available at `http://www.di.unito.it/~tutreeb/sentipolc-evalita16/index.html`

[9] After the deadline, SwissCheese and tweet2check reported about a conversion error from their internal format to the official one. The resubmitted amended runs are shown in the table (marked by the * symbol), but the official ranking was not revised.

Table 4: Task 1: F-scores for constrained ".c" and unconstrained runs ".u". After the deadline, two teams reported about a conversion error from their internal format to the official one. The resubmitted amended runs are marked with *.

System	Obj	Subj	F
Unitor.1.u	**0.6784**	**0.8105**	**0.7444**
Unitor.2.u	0.6723	0.7979	0.7351
samskara.1.c	0.6555	0.7814	**0.7184**
ItaliaNLP.2.c	0.6733	0.7535	0.7134
IRADABE.2.c	0.6671	0.7539	0.7105
INGEOTEC.1.c	0.6623	0.7550	0.7086
Unitor.c	0.6499	0.7590	0.7044
UniPI.1/2.c	**0.6741**	0.7133	0.6937
UniPI.1/2.u	0.6741	0.7133	0.6937
ItaliaNLP.1.c	0.6178	0.7350	0.6764
ADAPT.c	0.5646	0.7343	0.6495
IRADABE.1.c	0.6345	0.6139	0.6242
tweet2check16.c	0.4915	0.7557	0.6236
tweet2check14.c	0.3854	**0.7832**	0.5843
tweet2check14.u	0.3653	0.7940	0.5797
UniBO.1.c	0.5997	0.5296	0.5647
UniBO.2.c	0.5904	0.5201	0.5552
Baseline	0.0000	0.7897	0.3949
*SwissCheese.c_late	0.6536	0.7748	0.7142
*tweet2check16.u_late	0.4814	0.7820	0.6317

5.3 Task3: irony detection

Table 6 shows results for the irony detection task, which attracted 12 submissions from 7 teams. The highest F-score was achieved by tweet2check at 0.5412 (constrained run). The only unconstrained run was submitted by Unitor achieving 0.4810 as F-score. While all participating systems show an improvement over the baseline (F = 0.4688), many systems score very close to it, highlighting the complexity of the task.

6 Discussion

We compare the participating systems according to the following main dimensions: classification framework (approaches, algorithms, features), tweet representation strategy, exploitation of further Twitter annotated data for training, exploitation of available resources (e.g. sentiment lexicons, NLP tools, etc.), and issues about the interdependency of tasks in case of systems participating in several subtasks.

Since we did not receive details about the systems adopted by some participants, i.e., tweet2check, ADAPT and UniBO, we are not including them in the following discussion. We consider however tweet2check's results in the discussion regarding irony detection.

Approaches based on Convolutional Neural

Table 5: Task 2: F-scores for constrained ".c" and unconstrained runs ".u". Amended runs are marked with * .

System	Pos	Neg	F
UniPI.2.c	**0.6850**	0.6426	**0.6638**
Unitor.1.u	0.6354	**0.6885**	**0.6620**
Unitor.2.u	0.6312	0.6838	0.6575
ItaliaNLP.1.c	0.6265	**0.6743**	0.6504
IRADABE.2.c	0.6426	0.6480	0.6453
ItaliaNLP.2.c	0.6395	0.6469	0.6432
UniPI.1.u	**0.6699**	0.6146	0.6422
UniPI.1.c	0.6766	0.6002	0.6384
Unitor.c	0.6279	0.6486	0.6382
UniBO.1.c	0.6708	0.6026	0.6367
IntIntUniba.c	0.6189	0.6372	0.6281
IntIntUniba.u	0.6141	0.6348	0.6245
UniBO.2.c	0.6589	0.5892	0.6241
UniPI.2.u	0.6586	0.5654	0.6120
CoLingLab.c	0.5619	0.6579	0.6099
IRADABE.1.c	0.6081	0.6111	0.6096
INGEOTEC.1.u	0.5944	0.6205	0.6075
INGEOTEC.2.c	0.6414	0.5694	0.6054
ADAPT.c	0.5632	0.6461	0.6046
IntIntUniba.c	0.5779	0.6296	0.6037
tweet2check16.c	0.6153	0.5878	0.6016
tweet2check14.u	0.5585	0.6300	0.5943
tweet2check14.c	0.5660	0.6034	0.5847
samskara.1.c	0.5198	0.6168	0.5683
Baseline	0.4518	0.3808	0.4163
*SwissCheese.c_late	0.6529	0.7128	0.6828
*tweet2check16.u_late	0.6528	0.6373	0.6450

Table 6: Task 3: F-scores for constrained ".c" and unconstrained runs ".u". Amended runs are marked with *.

System	Non-Iro	Iro	F
tweet2check16.c	0.9115	0.1710	**0.5412**
CoMoDI.c	0.8993	0.1509	0.5251
tweet2check14.c	0.9166	0.1159	0.5162
IRADABE.2.c	0.9241	0.1026	0.5133
ItaliaNLP.1.c	0.9359	0.0625	0.4992
ADAPT.c	0.8042	**0.1879**	0.4961
IRADABE.1.c	0.9259	0.0484	0.4872
Unitor.2.u	**0.9372**	**0.0248**	**0.4810**
Unitor.c	0.9358	0.0163	0.4761
Unitor.1.u	0.9373	0.0084	0.4728
ItaliaNLP.2.c	**0.9367**	0.0083	0.4725
Baseline	0.9376	0.000	0.4688
*SwissCheese.c_late	0.9355	0.1367	0.5361

Networks (CNN) have been investigated at SENTIPOLC this year for the first time by a few teams. Most of the other teams adopted learning methods already investigated in SENTIPOLC 2014; in particular, Support Vector Machine (SVM) is the most adopted learning algorithm. The SVM is generally based over specific linguistic/semantic feature engineering, as discussed for example by ItaliaNLP, IRADABE, INGEOTEC or ColingLab. Other methods have been also used, as a Bayesian approach by samskara (achieving good results in polarity recognition) combined with linguistically motivated feature modelling. CoMoDi is the only participant that adopted a rule based approach in combination with a rich set of linguistic cues dedicated to irony detection.

Tweet representation schemas. Almost all teams adopted (i) traditional manual feature engineering or (ii) distributional models (i.e. Word embeddings) to represent tweets. The teams adopting the strategy (i) make use of traditional feature modeling, as presented in SENTIPOLC 2014, using specific features that encode word-based, syntactic and semantic (mostly lexicon-based) features. In addition, micro-blogging specific features such as emoticons and hashtags are also adopted, for example by ColingLab, INGEOTEC) or CoMoDi. Deep learning methods adopted by some teams, such as UniPi and SwissCheese required to model individual tweets through geometrical representation of tweets, i.e. vectors. Words from individual tweets are represented through Word Embeddings, mostly derived by using the Word2Vec tool or similar approaches. Unitor extends this representation with additional features derived from Distributional Polarity Lexicons. In addition, some teams (e.g. ColingLab) adopted Topic Models to represent tweets. Samskara also used feature modelling with a communicative and pragmatic value. CoMoDi is one of the few systems that investigated irony-specific features.

Exploitation of additional data for training. Some teams submitted unconstrained results, as they used additional Twitter annotated data for training their systems. In particular, UniPI used a silver standard corpus made of more than 1M tweets to pre-train the CNN; this corpus is annotated using a polarity lexicon and specific polarised words. Also Unitor used external tweets to pre-train their CNN. This corpus is made of the contexts of the tweets populating the training material and automatically annotated using the classifier trained only over the training material, in a semi-supervised fashion. Moreover, Unitor used distant supervision to label a set of tweets used for the acquisition of their so-called Distribution Polarity Lexicon. Distant supervision is also adopted by INGEOTEC to extend the training material for the their SVM classifier.

External Resources. The majority of teams used external resources, such as lexicons specific for Sentiment Analysis tasks. Some teams used already existing lexicons, such as Samskara, ItaliaNLP, CoLingLab, or CoMoDi, while others created their own task specific resources, such as Unitor, IRADABE, CoLingLab.

Issues about the interdependency of tasks. Among the systems participating in more than one task, SwissCheese and Unitor designed systems that exploit the interdependency of specific subtasks. In particular, SwissCheese trained one CNN for all the tasks simultaneously, by joining the labels. The results of their experiments indicate that the multi-task CNN outperforms the single-task CNN. Unitor made the training step dependent on the subtask, e.g. considering only subjective tweets when training the Polarity Classifier. However it is difficult to assess the contribution of cross-task information based only on the experimental results obtained by the single teams.

Irony detection. As also observed at SENTIPOLC 2014, irony detection appears truly challenging, as even the best performing system submitted by Tweet2Check ($F = 0.5412$) shows a low recall of 0.1710. We also observe that the performances of the supervised system developed by Tweet2Check and CoMoDi's rule-based approach, specifically tailored for irony detection, are very similar (Table 6).

While results seem to suggest that irony detection is the most difficult task, its complexity does not depend (only) on the inner structure of irony, but also on unbalanced data distribution (1 out of 7 examples is ironic in the training set). The classifiers are thus biased towards the non-irony class, and tend to retrieve all the non-ironic examples (high recall in the class non-irony) instead of actually modelling irony. If we measure the number of correctly predicted examples instead of the average of the two classes, the systems perform well (micro F1 of best system is 0.82).

Moreover, performance for irony detection drops significantly compared to SENTIPOLC 2014. An explanation for this could be that unlike SENTIPOLC 2014, at this edition the topics in the train and in the test sets are different, and it has been shown that systems might be modelling topic rather than irony (Barbieri et al., 2015). This evidence suggests that examples are probably not sufficient to generalise over the structure of ironic tweets. We plan to run further experiments on this issue, including a larger and more balanced dataset of ironic tweets in future campaigns.

7 Closing Remarks

All systems, except CoMoDI, exploited machine learning techniques in a supervised setting. Two main strategies emerged. One involves using linguistically principled approaches to represent tweets and provide the learning framework with valuable information to converge to good results. The other exploits state-of-the-art learning frameworks in combination with word embedding methods over large-scale corpora of tweets. On balance, the last approach achieved better results in the final ranks. However, with F-scores of 0.744 (unconstrained) and 0.7184 (constrained) in *subjectivity recognition* and 0.6638 (constrained) and 0.6620 (unconstrained) in *polarity recognition*, we are still far from having solved sentiment analysis on Twitter. For the future, we envisage the definition of novel approaches, for example by combining neural network-based learning with a linguistic-aware choice of features.

Besides modelling choices, *data* also matters. At this campaign we intentionally designed a test set with a sampling procedure that was close but not identical to that adopted for the training set (focusing again on political debates but on a different topic), so as to have a means to test the generalisation power of the systems (Basile et al., 2015). A couple of teams indeed reported substantial drops from the development to the official test set (e.g. IRADABE), and we plan to further investigate this aspect in future work. Overall, results confirm that sentiment analysis of micro-blogging is challenging, mostly due to the subjective nature of the phenomenon, and it's reflected in the inter-annotator agreement (Section 3.3). Crowdsourced data for this task also proved to be not entirely reliable, but this requires a finer-grained analysis on the collected data, and further experiments including a stricter implementation of the guidelines.

Although evaluated over different data, we see that this year's best systems show better, albeit comparable, performance for subjectivity with respect to 2014's systems, and outperform them for polarity (if we consider late submissions). For a proper evaluation across the various editions, we propose the use of a progress set for the next edition, as already done in the SemEval campaign.

References

Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. 2015. How Topic Biases Your Results? A Case Study of Sentiment Analysis and Irony Detection in Italian. In *RANLP, Recent Advances in Natural Language Processing*.

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on Italian tweets. In *Proc. of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta, Georgia, June.

Valerio Basile, Andrea Bolioli, Malvina Nissim, Viviana Patti, and Paolo Rosso. 2014. Overview of the Evalita 2014 SENTIment POLarity Classification Task. In *Proc. of EVALITA 2014*, pages 50–57, Pisa, Italy. Pisa University Press.

Pierpaolo Basile, Valerio Basile, Malvina Nissim, and Nicole Novielli. 2015. Deep tweets: from entity linking to sentiment analysis. In *Proc. of CLiC-it 2015*.

Pierpaolo Basile, Annalina Caputo, Anna Lisa Gentile, and Giuseppe Rizzo. 2016a. Overview of the EVALITA 2016 Named Entity rEcognition and Linking in Italian Tweets (NEEL-IT) Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proc. of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli. 2016b. EVALITA 2016: Overview of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In *Proc. of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Cristina Bosco, Viviana Patti, and Andrea Bolioli. 2013. Developing Corpora for Sentiment Analysis: The Case of Irony and Senti-TUT. *IEEE Intelligent Systems, Special Issue on Knowledge-based Approaches to Content-level Sentiment Analysis*, 28(2):55–63.

Cristina Bosco, Fabio Tamburini, Andrea Bolioli, and Alessandro Mazzei. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proc. of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Rebecca F. Bruce and Janyce M. Wiebe. 1999. Recognizing Subjectivity: A Case Study in Manual Tagging. *Nat. Lang. Eng.*, 5(2):187–205, June.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in Twitter and Amazon. In *Proc. of CoNLL '10*, pages 107–116.

Aniruddha Ghosh, Guofu Li, Tony Veale, Paolo Rosso, Ekaterina Shutova, Antonio Reyes, and Jhon Barnden. 2015. Semeval-2015 task 11: Sentiment analysis of figurative language in Twitter. In *Proc. of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 470–475.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: A closer look. In *Proc. of ACL–HLT '11*, pages 581–586, Portland, Oregon.

Yanfen Hao and Tony Veale. 2010. An ironic fist in a velvet glove: Creative mis-representation in the construction of ironic similes. *Minds Mach.*, 20(4):635–650.

Anne-Lyse Minard, Manuela Speranza, and Tommaso Caselli. 2016. The EVALITA 2016 Event Factuality Annotation Task (FactA). In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proc. of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2016. Semeval-2016 task 4: Sentiment analysis in twitter. In *Proc. of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1–18.

Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, January.

Antonio Reyes and Paolo Rosso. 2014. On the difficulty of automatically detecting irony: Beyond a simple case of negation. *Knowl. Inf. Syst.*, 40(3):595–614.

Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Lang. Resour. Eval.*, 47(1):239–268, March.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. SemEval-2014 Task 9: Sentiment Analysis in Twitter. In *Proc. of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland, August.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif M Mohammad, Alan Ritter, and Veselin Stoyanov. 2015. SemEval-2015 Task 10: Sentiment Analysis in Twitter. In *Proc. of the 9th International Workshop on Semantic Evaluation*, SemEval '2015.

Marco Stranisci, Cristina Bosco, Delia Iraz Hernndez Faras, and Viviana Patti. 2016. Annotating sentiment and irony in the online italian political debate on #labuonascuola. In *Proc. of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2892–2899. ELRA.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 1(2).

Convolutional Neural Networks for Sentiment Analysis on Italian Tweets

Giuseppe Attardi, Daniele Sartiano, Chiara Alzetta, Federica Semplici
Dipartimento di Informatica
Università di Pisa
Largo B. Pontecorvo, 3
I-56127 Pisa, Italy
{attardi, sartiano}@di.unipi.it,
{c.alzetta, f.semplici}@studenti.unipi.it

Abstract

English. The paper describes our submission to the task 2 of SENTIment POLarity Classification in Italian Tweets at Evalita 2016. Our approach is based on a convolutional neural network that exploits both word embeddings and Sentiment Specific word embeddings. We also experimented a model trained with a distant supervised corpus. Our submission with Sentiment Specific word embeddings achieved the first official score.

Italiano. *L'articolo descrive la nostra partecipazione al task 2 di SENTIment POLarity Classification in Italian Tweets a Evalita 2016. Il nostro approccio si basa su una rete neurale convoluzionale che sfrutta sia word embeddings tradizionali che sentiment specific word embeddings. Abbiamo inoltre sperimentato un modello allenato su un corpus costruito mediante tecnica distant supervised. Il nostro sistema, che utilizza Specific Sentiment word embeddings, ha ottenuto il primo punteggio officiale.*

1 Introduction

The paper describes our submissions to the Task 2 of SENTiment POLarity Classification at Evalita 2016 (Barbieri et al. 2016).

In Sentipolc the focus is the sentiment analysis of the in Italian tweets, it is divided in three subtasks:

- Task 1: Subjectivity Classification: identify the subjectivity of a tweet.
- Task 2: Polarity Classification: classify a tweet as positive, negative, neutral or mixed (i.e. a tweet with positive and negative sentiment).
- Task 3: Irony Detection: identify if is present the irony in a tweet.

The state of the art on the polarity classification of tweets is the application of Deep Learning methods (Nakov et al., 2016), like convolutional neural network or recurrent neural networks, in particular long short-term memory networks (Hochreiter, and Schmidhuber, 1997).

We explored Deep Learning techniques for the sentiment analysis of English tweets at Semeval 2016 with good results, where we noticed that use of convolutional neural network and Sentiment Specific word embeddings was promising.

We applied a similar approach for the Italian language, building word embeddings from a big corpus of Italian tweets, sentiment specific word embeddings from positive and negative tweets, using a convolutional neural network as classifier. We also introduced a distant supervised corpus as silver training set.

We report the results of our experiments with this approach on the task Evalita 2016 Sentipolc Task 2 Polarity classification.

2 Description of the System

The architecture of the system consists of the following steps:

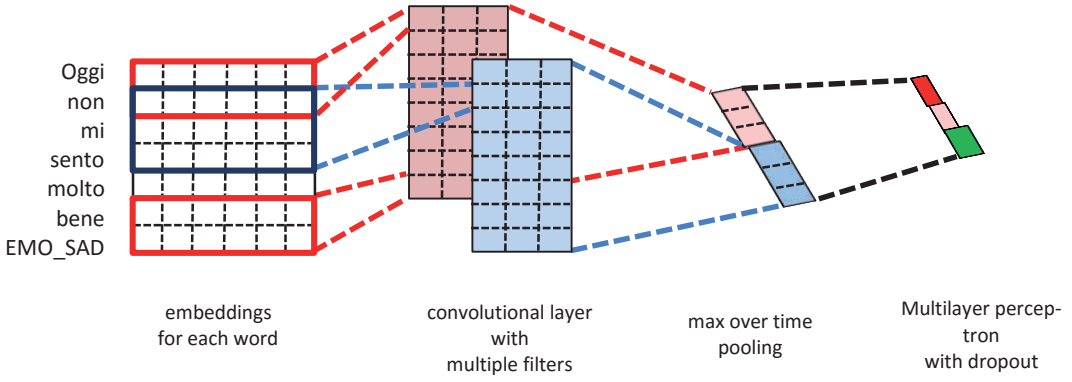


Figure 1. The Deep Learning classifier.

- build word embeddings from a collection of 167 million tweets collected with the Twitter API over a period of May to September 2016, preprocessed as described later.
- build Sentiment Specific word embeddings using a portion of these tweets split into positive/negative by distant supervision.
- train a convolutional neural network classifier using one of the above word embeddings

The convolutional neural network classifier exploits pre-trained word embeddings as only features in various configurations as described below. The architecture of the classifier consists of the following layers described in Figure 1: a lookup layer for word embeddings, a convolutional layer with a ReLU activation function, a maxpooling layer, a dropout layer, a linear layer with tanh activation and a softmax layer. This is the same classifier described in (Attardi and Sartiano, 2016), that achieved good results at the SemEval 2016 task 4 on Sentiment Analysis in Twitter (Nakov et al., 2016). Here we test it on a similar task for Italian tweets.

2.1 Data Preprocessing

In order to build the word embeddings we preprocessed the tweets using tools from the Tanl pipeline (Attardi et al., 2010): the sentence splitter and the specialized tweet tokenizer for the tokenization and the normalization of tweets. Normalization involved replacing the mentions with the string "@mention", emoticons with their name (e.g. "EMO_SMILE") and URLs with "URL_NORM".

2.2 Word Embeddings and Sentiment Specific Word Embeddings

We experimented with standard word embeddings, in particular building them with the tool word2vec[1] (Mikolov, 2013), using the skip-gram model. These word embeddings though do not take into account semantic differences between words expressing opposite polarity, since they basically encode co-occurrence information as shown by (Levy and Goldberg, 2014). For encodes sentiment information in the continuous representation of words, we use the technique of Tang et al. (2014) as implemented in the DeepNL[2] library (Attardi, 2015). A neural network with a suitable loss function provides the supervision for transferring the sentiment polarity of texts into the embeddings from generic tweets.

2.3 Distant supervision

The frequency distribution of classes in the dataset, as shown in Table 1, seems skewed and not fully representative of the distribution in a statistical sample of tweets: negative tweets are normally much less frequent than positive or neutral ones (Bravo-Marquez, 2015). To reduce this bias and to increase the size of the training set, we selected additional tweets from our corpus of Italian tweets by means distant supervision. In the first step we selected the tweets belonging to a class (positive, negative, neutral, mixed) via regular expressions. In the second step, the selected tweets are classified by the classifier trained using the task trainset. The silver corpus is built taking the tweets with the matched class between the regular expression system and the classifier.

[1] https://code.google.com/archive/p/word2vec/
[2] https://github.com/attardi/deepnl

3 Experiments

The plain word embeddings were built applying vord2vec to a collection of 167 million Italian unlabeled tweets, using the the skip gram model, and the following parameters: embeddings size 300, window dimension 5, discarding word that appear less than 5 times. We obtained about 450k word embeddings.

The sentiment specific word embeddings (SWE) were built with DeepNL, starting from the word embeddings built at the previous step and tuning them with a supervised set of positive or negative tweets, obtained as follows from 2.3 million tweets selected randomly from our corpus of collected tweets:

- Positive tweet: one that contains only emoticons from a set of positive emoticons (e.g. smiles, hearts, laughs, expressions of surprise, angels and high fives).
- Negative tweet: one that contains only emoticons from a set of negative emotions (e.g. tears, angry and sad).

Integris srl cooperated to the task providing a set of 1.3 million tweets, selected by relying on a lexicon of handcrafted polarized words. This resource is also added to the corpus.

We split the training set provided for the Evalita 2016 SentiPolc Task into a train set (5335 tweets), a validation set (592 tweets) and a test set (1482 tweets). This dataset was tokenized and normalized as described in Section 2.1.

For the take of participating to subtask 2, polarity classification, the 13-value annotations present in the datasets were converted into four values: "neutral", "positive", "negative" and "mixed" depending on the values of the fields "opos" and "oneg", which express the tweet polarity, according to the task guidelines[3]. We did not take into account the values for "lpos" and "lneg".

The frequency distribution of these classes turns out to be quite unbalanced, as shown in Table 1.

Class	Train set	Validation set
Neutral	2262	554
Negative	2029	513
Positive	1299	312
Mixed	337	103

Table 1. Task dataset distribution

[3] http://www.di.unito.it/~tutreeb/sentipolc-evalita16/sentipolc-guidelines2016UPDATED130916.pdf

The training set is still fairly small, compared for example to the size of the corpus used in SemEval 2106. The "mixed" class in particular is small in absolute numbers, even though not in percentage value, which makes hard to properly train a ML classifier.

Therefore we tried to increase the training set by means of the distant supervision as described above: we selected a maximum of 10,000 tweets for class via regular expressions, then we classified them with the classifier trained with the gold training set. We chose for addition into a silver training set, the tweets which were assigned by the classifier the same class of the regular expression. As reported in Table 2, the silver dataset remains unbalanced; in particular, no "mixed" example was added to the original train set.

Class	Train set	Dev set
Neutral	8505	554
Negative	5987	513
Positive	6813	312
Mixed	337	103

Table 2. Distant supervised dataset distribution.

Table 3 shows the common settings used for training the classifier. We used the same parameters as SemEval-2016.

Word Embeddings Size	300
Hidden Units	100
Dropout Rate	0.5
Batch size	50
Adadelta Decay	0.95
Epochs	50

Table 3. Network Common Settings

We performed extensive experiments with the classifier in various configurations, varying the number of filters; the use of skip-gram word embeddings or sentiment specific word embeddings; different training sets, either the gold one or the silver one. Results of the evaluation on the validation set allowed us to choose the best settings, as listed in the Table 4. Best Settings.

	Run1		Run2	
Embeddings	WE skipgram		SWE	
Training set	Gold	Silver	Gold	Silver
Filters	2,3,5	4,5,6,7	7,7,7,7,8,8,8,8	7,8,9,10

Table 4. Best Settings

4 Results

We submitted four runs for the subtask 2 "polarity classification":

- UniPI_1.c: gold training set, word embeddings with skip-gram model, filters: "2,3,5".
- UniPI_1.u: silver corpus as training set, word embeddings with skip-gram model, filters: "4,5,6,7".
- UniPI_2.c: gold training set, sentiment specific word embeddings, filters: "7,7,7,7,8,8,8,8".
- UniPI_2.u: silver corpus as training set, sentiment specific word embeddings, filters: "7,8,9,10".

The following table reports the top official results for the subtask 2:

System	Positive F-score	Negative F-score	Combined F-score
UniPI_2.c	**0.685**	0.6426	**0.6638**
team1_1.u	0.6354	**0.6885**	0.662
team1_2.u	0.6312	0.6838	0.6575
team4_.c	0.644	0.6605	0.6522
team3_.1.c	0.6265	0.6743	0.6504
team5_2.c	0.6426	0.648	0.6453
team3_.2.c	0.6395	0.6469	0.6432
UniPI_1.u	0.6699	0.6146	0.6422
UniPI_1.c	0.6766	0.6002	0.6384
UniPI_2.u	0.6586	0.5654	0.612

Table 5. Top official results for SentiPolc subtask 2.

The run UniPI_2.c achieved the top overall score among a total of 26 submissions to task 2. This confirms the effectiveness of sentiment specific word embeddings in sentiment polarity classification also for Italian tweets.

The use of an extended silver corpus did not provide significant benefits, possibly because the resulting corpus was still unbalanced.

In addition to the subtask 2, we submitted one run for the Task 1 "Subjectivity Classification": given a message, decide whether the message is subjective or objective. We used the same classifier for the subtask 2, using only two classes (subjective, objective), with the same skip-gram word embeddings used for the other task and the configuration listed in Table 3, using the following filters: "7,8,9,10", without performing extensive experiments. The following table reports the top official results for the subtask 1:

system	Objective F-score	Subjective F-score	Combined F-score
team1_1.u	**0.6784**	**0.8105**	**0.7444**
team1_2.u	0.6723	0.7979	0.7351
team2_.1.c	0.6555	0.7814	0.7184
team3_.2.c	0.6733	0.7535	0.7134
team4_.c	0.6465	0.775	0.7107
team5_2.c	0.6671	0.7539	0.7105
team6_.c	0.6623	0.755	0.7086
team1_.c	0.6499	0.759	0.7044
UniPI_1	0.6741	0.7133	0.6937
team3_.1.c	0.6178	0.735	0.6764
team8_.c	0.5646	0.7343	0.6495
team5_1.c	0.6345	0.6139	0.6242

Table 6 Top official results for SentiPolc subtask 1.

5 Discussion

We confirmed the validity of the convolutional neural networks in the twitter sentiment classification, also for the Italian language.

The system achieved top score in the task 2 of SENTiment POLarity Classification Task of Evalita 2016.

Acknowledgments

We gratefully acknowledge the support of the University of Pisa through project PRA and NVIDIA Corporation for a donation of the Tesla K40 GPU used in the experiments.

Integris srl cooperated by providing a corpus of sentiment annotated tweets.

References

Giuseppe Attardi, Stefano Dei Rossi, and Maria Simi. 2010. *The Tanl Pipeline*. In Proc. of LREC Workshop on WSPP, Malta.

Giuseppe Attardi. 2015. DeepNL: a Deep Learning NLP pipeline. *Workshop on Vector Space Modeling for NLP*. Proc. of NAACL HLT 2015, Denver, Colorado (June 5, 2015).

Giuseppe Attardi and Daniele Sartiano. 2016. UniPI at SemEval-2016 Task 4: Convolutional Neural Networks for Sentiment Classification. *Proceedings of SemEval*, 220-224.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian* (EVALITA 2016).

Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. 2015. Positive, negative, or neutral: Learning an expanded opinion lexicon from emoticon-annotated tweets. *IJCAI 2015*. AAAI Press.

Sepp Hochreiter, and Jürgen Schmidhuber. (1997). Long short-term memory. *Neural computation 9.8* 1735-1780.

Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. In *Advances in neural information processing systems*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*. arXiv:1310.4546

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2016. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th international workshop on semantic evaluation* (SemEval 2016), San Diego, US (forthcoming).

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014, June. Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification. In *ACL (1)* (pp. 1555-1565).

IRADABE2: Lexicon Merging and Positional Features for Sentiment Analysis in Italian

Davide Buscaldi
LIPN, Université Paris 13
Villetaneuse, France
buscaldi@lipn.univ-paris13.fr

Delia Irazú Hernandez-Farias
Dipartimento di Informatica
Università degli studi di Torino
Turin, Italy

PRHLT group
Universitat Politècnica de València
Valencia, Spain
dhernandez1@dsic.upv.es

Abstract

English. This paper presents the participation of the IRADABE team to the SENTIPOLC 2016 task. This year we investigated the use of positional features together with the fusion of sentiment analysis resources with the aim to classify Italian tweets according to subjectivity, polarity and irony. Our approach uses as starting point our participation in the SENTIPOLC 2014 edition. For classification we adopted a supervised approach that takes advantage of support vector machines and neural networks.

Italiano. *Quest'articolo presenta il lavoro svolto dal team IRADABE per la partecipazione al task SENTIPOLC 2016. Il lavoro svolto include l'utilizzo di caratteristiche posizionali e la fusione di lessici specialistici, finalizzato alla classificazione di tweet in italiano, secondo la loro soggettività, polarità ed ironia. Il nostro approccio si basa sull'esperienza acquisita nel corso della partecipazione all'edizione 2014 di SENTIPOLC. Per la classificazione sono stati adottati dei metodi supervisionati come le macchine a supporto vettoriale e le reti neurali.*

1 Introduction

Sentiment analysis (SA) related tasks have attracted the attention of many researchers during the last decade. Several approaches have been proposed in order to address SA. Most of them have in common the use of machine learning together with natural language processing techniques. Despite all those efforts there still many challenges left such as: multililngual sentiment analysis, i.e, to perform SA in languages different from English (Mohammad, 2016). This year for the second time a sentiment analysis task on Italian tweets has been organized at EvalIta, the Sentiment Polarity Classification (SENTIPOLC) task (Barbieri et al., 2016).

In this paper we study the effect of positional features over the sentiment, irony and polarity classification tasks in the context of SENTIPOLC 2016 task. We propose a revised version of our IRADABE system (Hernandez-Farias et al., 2014), which participated with fairly good results in 2014. The novelties for this participation are not only in the positional features, but also in a new sentiment lexicon that was built combining and expanding the lexicons we used in 2014.

The rest of the paper is structured as follows: in Section 2 we describe the steps we took to build an enhanced sentiment dictionary in Italian from existing English resources; in Section 3 we describe the new positional features of the IRADABE system.

2 Building a unified dictionary

In sentiment analysis related tasks, there are several factors that can be considered in order to determine the polarity of a given piece of text. Overall, the presence of positive or negative words is used as a strong indicator of sentiment. Nowadays there are many sentiment analysis related resources that can be exploited to infer polarity from texts. Recently, this kind of lexicons has been proven to be effective for detecting irony in Twitter (Hernańdez Farías et al., 2016). Unfortunately, the majority of available resources are in English. A common practice to deal with the lack of resources in different languages is to automatically translate it from English.

However, the language barrier is not the only drawback for these resources. Another issue is

the limited coverage of certain resources. For instance, AFINN (Nielsen, 2011) includes only 2477 words in its English version, and the Hu-Liu lexicon (Hu and Liu, 2004) contains about 6800 words. We verified on the SENTIPOLC14 training set that the Hu-Liu lexicon provided a score for 63.1% of training sentences, while the coverage for AFINN was of 70.7%, indicating that the number of items in the lexicons is not proportional to the expected coverage; in other words, although AFINN is smaller, the words included are more frequently used than those listed in the Hu-Liu lexicon. The coverage provided by a hypothetical lexicon obtaining by the combination of the two resources would be 79.5%.

We observed also that in some cases these lexicons provide a score for a word but not for one of their synonyms: in the Hu-Liu lexicon, for instance, the word 'repel' is listed as a negative one, but 'resist', which is listed as one of its synonym in the Roget's thesaurus[1], is not. SentiWordNet (Baccianella et al., 2010) compensates some of the issues; its coverage is considerably higher than the previously named lexicons: 90.6% on the SENTIPOLC14 training set. Its scores are also assigned to synsets, and not words. However, it is not complete: we measured that a combination of SentiWordNet with AFINN and Hu-Liu would attain a coverage of 94.4% on the SENTIPOLC14 training set. Moreover, the problem of working with synsets is that it is necessary to carry out word sense disambiguation, which is a difficult task, particularly in the case of short sentences like tweets. For this reason, our translation of SentiWordNet into Italian (Hernandez-Farias et al., 2014) resulted in a word-based lexicon and not a synset-based one.

Therefore, we built a sentiment lexicon which was aimed to provide the highest possible coverage by merging existing resources and extending the scores to synonyms or quasi-synonyms. The sentiment lexicon was built following a three-step process:

1. Create a unique set of opinion words from the AFINN, Hu-Liu and SentiWordNet lexicons, and merge the scores if multiple scores are available for the same word; the original English resources were previously translated into the Italian language for our participation in SENTIPOLC 2014;

2. Extend the lexicon with the WordNet synonyms of words obtained in step 1;

3. Extend the lexicon with pseudo-synonyms of words obtained in step 1 and 2, using word2vec for similarity. We denote them as "pseudo-synonyms" because the similarity according to word2vec doesn't necessarily means that the words are synonyms, only that they usually share the same contexts.

The scores at each step were calculated as follows: in step 1, the weight of a word is the average of the non-zero scores from the three lexicons. In step 2, the weight for a synonym is the same of the originating word. If the synonym is already in the lexicon, then we keep the most polarizing weight (if the scores have the same sign), or the sum of the weights (if the scores have opposed signs). For step 3 we previously built semantic vectors using word2vec (Mikolov et al., 2013) on the ItWaC[2] corpus (Baroni et al., 2009). Then, we select for each word in the lexicon obtained at step 2 the 10 most similar pseudo-synonyms having a similarity score ≥ 0.6. If the related pseudo-synonym already exists in the lexicon, its score is kept, otherwise it is added to the lexicon with a polarity resulting from the score of the original word multiplied by the similarity score of the pseudo-synonym. We named the obtained resource the 'Unified Italian Semantic Lexicon', shortened as UnISeLex. It contains $31,601$ words. At step 1, the dictionary size was $12,102$; at step 2, after adding the synonyms, it contained $15,412$ words.

In addition to this new resource, we exploited *labMT-English words*. It is a list (Dodds et al., 2011) composed of 10,000 words manually annotated with a happiness measure in a range between 0 up to 9. These words were collected from different resources such as Twitter, Google Books, music lyrics, and the New York Times (1987 to 2007).

3 Positional Features

It is well known that in the context of opinion mining and summarization the position of opinion words is an important feature (Pang and Lee, 2008), (Taboada and Grieve, 2004). In reviews,

[1] http://www.thesaurus.com/Roget-Alpha-Index.html

[2] http://wacky.sslmit.unibo.it

users tend to summarize the judgment in the final sentence, after a comprehensive analysis of the various features of the item being reviewed (for instance, in a movie review, they would review the photography, the screenplay, the actor performance, and finally provide an overall judgment of the movie). Since SENTIPOLC is focused on tweets, whose length is limited to 140 characters, there is less room for a complex analysis and therefore it is not clear whether the position of sentiment words is important or not.

In fact, we analyzed the training set and noticed that some words tend to appear in certain positions when the sentence is labelled with a class rather than the other one. For example, in the subjective sub-task, 'non' (not), 'io' (I), auxiliary verbs like 'potere' (can), 'dovere' (must) tend to occur mostly at the beginning of the sentence if the sentence is subjective. In the positive polarity subtask, words like 'bello' (beautiful), 'piacere' (like) and 'amare' (love) are more often observed at the beginning of the sentence if the tweet is positive.

We therefore introduced a positional Bag-of-Words (BOW) weighting, where the weight of a word t is calculated as:

$$w(t) = 1 + pos(t)/len(s)$$

where $pos(t)$ is the *last* observed position of the word in the sentence, and $len(s)$ is the length of the sentence. For instance, in the sentence "I *love* apples in fall.", $w(love) = 1 + 1/5 = 1.2$, since the word *love* is at position 1 in a sentence of 5 words.

The Bag of Words was obtained by taking all the lemmatized forms w that appeared in the training corpus with a frequency greater than 5 and $I(w) > 0.001$, where $I(w)$ is the informativeness of word w calculated as:

$$I(w) = p(w|c^+)\left(\log(p(w|c^+)) - \log(p(w|c^-))\right)$$

where $p(w|c^+)$ and $p(w|c^-)$ are the probabilities of a word appearing in the tweets tagged with the positive or negative class, respectively. The result of this selection consisted in 943 words for the $subj$ subtask, 831 for pos, 991 for neg and 1197 for iro.

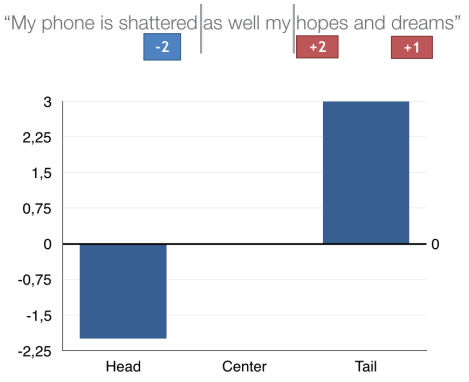
The results in Table 3 show a marginal improvement for the polarity and irony classes, while in subjectivity the system lost 2% in F-measure. This is probably due to the fact that the important words that tend to appear in the first part of the sentence may repeat later, providing a wrong score for the feature.

	Subj	***Pol(+)***	***Pol(-)***	***Iro***
pos. BOW	0.528	0.852	0.848	0.900
std. BOW	0.542	0.849	0.842	0.894

Table 1: F-measures for positional and standard BOW models trained on the train part of the dev set; results are calculated on the test part of the dev set.

With respect to the 2014 version of IRADABE, we introduced 3 more position-dependent features. Each tweet was divided into 3 sections, *head*, *centre* and *tail*. For each section, we consider the sum of the sentiment scores of the included words as a separate feature. Therefore, we have three features, named in Table 3.1 as $headS$, $centreS$ and $tailS$.

Figure 1: Example of lexicon positional scores for the sentence "My phone is shattered as well my hopes and dreams".



3.1 Other features

We renewed most of the features used for SENTIPOLC 2014, with the main difference that we are now using a single sentiment lexicon instead than 3. In IRADABE 2014 we grouped the features into two categories: *Surface Features* and *Lexicon-based Features*. We recall the ones appearing in Table 2, directing the reader to (Hernandez-Farias et al., 2014) for a more detailed description. The first group comprises features such as the presence of an URL address ($http$), the length of the tweet ($length$), a list of swearing words ($taboo$), and the ratio of uppercase characters ($shout$). Among the features extracted from dictionaries, we used the sum of polarity scores ($polSum$), the sum of only negative or pos-

itive scores ($sum(-)$ and $sum(+)$), the number of negative scores ($count(-)$) on UnISeLex, and the average and the standard deviation of scores on labMT (avg_{labMT} and std_{labMT}, respectively). Furthermore, to determine both polarity and irony, a subjectivity indicator (*subj*) feature was used; it is obtained by identifying first if a tweet is subjective or not. Finally, the $mixed$ feature indicates is the tweet has mixed polarity or not.

Subj	***Pol(+)***	***Pol(-)***	***Iro***
$http$	$subj$	$subj$	$subj$
$shout$	avg_{labMT}	$sum(-)$	$http$
$sum(-)$	'grazie'	$count(-)$	'governo'
$count(-)$	$smileys$	avg_{labMT}	$mixed$
$headS$	$polSum$	$length$	$shout$
$pers$	$http$	$polSum$	'Mario'
'!'	'?'	$http$	'che'
avg_{labMT}	$sum(+)$	$centreS$	'#Grillo'
'mi'	'bello'	$taboo$	$length$
$taboo$	'amare'	std_{labMT}	$sum(-)$

Table 2: The 10 best features for each subtask in the training set.

4 Results and Discussion

We evaluated our approach on the dataset provided by the organizers of SENTIPOLC 2016. This dataset is composed by up to 10,000 tweets distributed in training set and test set. Both datasets contain tweets related to political and socio-political domains, as well as some generic tweets[3].

We experimented with different configurations for assessing subjectivity, polarity and irony. We sent two runs for evaluation purposes in SENTIPOLC-2016:

- *run 1*. For assessign the subjectivity label a Tensorflow[4] implementation of Deep Neural Ngetwork (DNN) was applied, with 2 hidden layers with 1024 and 512 states, respectively. Then, the polarity and irony labels were determined by exploiting a SVM[5].

- *run 2*. In this run, the bag-of-words were revised to remove words that may have a different polarity depending on the context (. Classification was carried out using a SVM (radial basis function kernel) for all subtasks, including $subj$.

From the results, we can observe that the DNN obtained an excellent precision (more than 93%) in $subj$, but the recall was very low. This may indicate a problem due to the class not being balanced, or an overfitting problem with the DNN, which is plausible given the number of features. This may also be the reason for which the SVM performs better, because SVMs are less afflicted by the "curse of dimensionality".

run 1				
	Subj	***Pol(+)***	***Pol(-)***	***Iro***
Precision	0.9328	0.6755	0.5161	0.1296
Recall	0.4575	0.3325	0.2273	0.0298
F-Measure	0.6139	0.4456	0.3156	0.0484
run 2				
	Subj	***Pol(+)***	***Pol(-)***	***Iro***
Precision	0.8714	0.6493	0.4602	0.2078
Recall	0.6644	0.4377	0.3466	0.0681
F-Measure	0.7539	0.5229	0.3955	0.1026

Table 3: Official results of our model on the test set.

5 Conclusions

As future work, it could be interesting to exploit the labels for exact polarity as provided by the organizers. This kind of information could help in some way to identify the use of figurative language. Furthermore, we are planning to enrich IRADABE with other kinds of features that allow us to cover more subtle aspects of sentiment, such as emotions. The introduction of the "happiness score" provided by labMT was particularly useful, with the related features being critical in the subjectivity and polarity subtasks. This motivates us to look for dictionaries that may express different feelings than just the overall polarity of a word. We will also need to verify the effectiveness of the resource we produced automatically with respect to other hand-crafted dictionaries for the Italian language, such as Sentix (Basile and Nissim, 2013)

We plan to use a more refined weighting scheme for the positional features, such as the locally-weighted bag-of-words or LOWBOW (Lebanon et

[3]Further details on the datasets can be found in the task overview (Barbieri et al., 2016)

[4]http://www.tensorflow.org

[5]As in IRADABE-2014 version, the subjectivity label influences the determination of both the polarity values and the presence of irony.

al., 2007), although it would mean an increase of the feature space of at least 3 times (if we keep the head, centre, tail cuts), probably furtherly compromising the use of DNN for classification.

About the utility of positional features, the current results are inconclusive, so we need to investigate further about how the positional scoring affects the results. On the other hand, the results show that the merged dictionary was a useful resource, with dictionary-based features representing 25% of the most discriminating features.

Acknowledgments

This research work has been supported by the "Investissements d'Avenir" program ANR-10-LABX-0083 (Labex EFL). The National Council for Science and Technology (CONACyT Mexico) has funded the research work of Delia Irazú Hernández Faríias (Grant No. 218109/313683 CVU-369616).

References

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pages 2200,2204, Valletta, Malta, may. European Language Resources Association (ELRA).

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, Eros Zanchetta, Springer, and Science+business Media B. V. 2009. The wacky wide web: A collection of very large linguistically processed webcrawled corpora. language resources and evaluation.

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on Italian tweets. In *WASSA 2013*, Atlanta, United States, June.

Peter Sheridan Dodds, Kameron Decker Harris, Isabel M. Kloumann, Catherine A. Bliss, and Christopher M. Danforth. 2011. Temporal patterns of happiness and information in a global social network: Hedonometrics and twitter. *PLoS ONE*, 6(12).

Irazú Hernandez-Farias, Davide Buscaldi, and Belém Priego-Sánchez. 2014. IRADABE: Adapting English Lexicons to the Italian Sentiment Polarity Classification task. In *First Italian Conference on Computational Linguistics (CLiC-it 2014) and the fourth International Workshop EVALITA2014*, Proceedings of the First Italian Conference on Computational Linguistics (CLiC-it 2014) and the fourth International Workshop EVALITA2014, pages 75–81, Pisa, Italy, December.

Delia Irazú Hernańdez Farías, Viviana Patti, and Paolo Rosso. 2016. Irony detection in twitter: The role of affective content. *ACM Trans. Internet Technol.*, 16(3):19:1–19:24, July.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, Seattle, WA, USA. ACM.

Guy Lebanon, Yi Mao, and Joshua Dillon. 2007. The locally weighted bag of words framework for document representation. *Journal of Machine Learning Research*, 8(Oct):2405–2441.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Saif M. Mohammad. 2016. Challenges in sentiment analysis. In *A Practical Guide to Sentiment Analysis*. Springer.

Finn Årup Nielsen. 2011. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*, volume 718 of *CEUR Workshop Proceedings*, pages 93–98, Heraklion, Crete, Greece. CEUR-WS.org.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

Maite Taboada and Jack Grieve. 2004. Analyzing appraisal automatically. In *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*, pages 158–161, Stanford, US.

Context-aware Convolutional Neural Networks for Twitter Sentiment Analysis in Italian

Giuseppe Castellucci, Danilo Croce, Roberto Basili
Department of Enterprise Engineering
University of Roma, Tor Vergata
Via del Politecnico 1, 00133 Roma, Italy
castellucci@ing.uniroma2.it, {croce,basili}@info.uniroma2.it

Abstract

English. This paper describes the Unitor system that participated to the *SENTIment POLarity Classification* task proposed in Evalita 2016. The system implements a classification workflow made of several Convolutional Neural Network classifiers, that generalize the linguistic information observed in the training tweets by considering also their context. Moreover, sentiment specific information is injected in the training process by using Polarity Lexicons automatically acquired through the automatic analysis of unlabeled collection of tweets. Unitor achieved the best results in the Subjectivity Classification sub-task, and it scored 2^{nd} in the Polarity Classification sub-task, among about 25 different submissions.

Italiano. *Questo lavoro descrive il sistema Unitor valutato nel task di SENTIment POLarity Classification proposto all'interno di Evalita 2016. Il sistema é basato su un workflow di classificazione implementato usando Convolutional Neural Network, che generalizzano le evidenze osservabili all'interno dei dati di addestramento analizzando i loro contesti e sfruttando lessici specifici per la analisi del sentimento, generati automaticamente. Il sistema ha ottenuto ottimi risultati, ottenendo la miglior performance nel task di Subjectivity Classification e la seconda nel task di Polarity Classification.*

1 Introduction

In this paper, the Unitor system participating in the *Sentiment Polarity Classification* (SENTIPOLC) task (Barbieri et al., 2016) within the Evalita 2016 evaluation campaign is described. The system is based on a cascade of three classifiers based on Deep Learning methods and it has been applied to all the three sub-tasks of SENTIPOLC: *Subjectivity Classification*, *Polarity Classification* and the pilot task called *Irony Detection*. Each classifier is implemented with a Convolutional Neural Network (CNN) (LeCun et al., 1998) according the modeling proposed in (Croce et al., 2016). The adopted solution extends the CNN architecture proposed in (Kim, 2014) with (i) sentiment specific information derived from an automatically derived polarity lexicon (Castellucci et al., 2015a), and (ii) with the contextual information associated with each tweet (see (Castellucci et al., 2015b) for more information about the contextual modeling in SA in Twitter). The Unitor system ranked 1^{st} in the Subjectivity Classification task and 2^{nd} in the Polarity Detection task among the unconstrained systems, resulting as one of the best solution in the challenge. It is a remarkable result as the CNNs have been trained without any complex feature engineering but adopting almost the same modeling in each sub-task. The proposed solution allows to achieve state-of-the-art results in *Subjectivity Classification* and *Polarity Classification* task by applying unsupervised analysis of unlabeled data that can be easily gathered by Twitter.

In Section 2 the deep learning architecture adopted in Unitor is presented, while the classification workflow is presented in 3. In Section 4 the experimental results are reported and discussed, while Section 5 derives the conclusions.

2 A Sentiment and Context aware Convolutional Neural Networks

The Unitor system is based on the Convolutional Neural Network (CNN) architecture for text classification proposed in (Kim, 2014), and further extended in (Croce et al., 2016). This deep net-

work is characterized by 4 layers (see Figure 1).

The *first layer* represents the input through word embedding: it is a low-dimensional representation of words, which is derived by the unsupervised analysis of large-scale corpora, with approaches similar to (Mikolov et al., 2013). The embedding of a vocabulary V is a look-up table $\mathbf{E}$, where each element is the $d-$dimensional representation of a word. Details about this representation will be discussed in the next sections. Let $\mathbf{x}_i \in \mathbb{R}^d$ be the d-dimensional representation of the i-th word. A sentence of length n is represented through the concatenation of the word vectors composing it, i.e., a matrix $\mathbf{I}$ whose dimension is $n \times d$.

The *second layer* represents the convolutional features that are learned during the training stage. A *filter*, or *feature detector*, $\mathbf{W} \in \mathbb{R}^{f \times d}$, is applied over the input layer matrix producing the learned representations. In particular, a new feature c_i is learned according to: $c_i = g(\mathbf{W} \cdot \mathbf{I}_{i:i+f-1} + b)$, where g is a non-linear function, such as the rectifier function, $b \in \mathbb{R}$ is a bias term and $\mathbf{I}_{i:i+f-1}$ is a portion of the input matrix along the first dimension. In particular, the filter slides over the input matrix producing a feature map $\mathbf{c} = [c_1, \ldots, c_{n-h+1}]$. The filter is applied over the whole input matrix by assuming two key aspects: *local invariance* and *compositionality*. The former specifies that the filter should learn to detect patterns in texts without considering their exact position in the input. The latter specifies that each local patch of height f, i.e., a f-gram, of the input should be considered in the learned feature representations. Ideally, a f-gram is composed through $\mathbf{W}$ into a higher level representation.

In practice, multiple filters of different heights can be applied resulting in a set of learned representations, which are combined in a *third layer* through the *max-over-time* operation, i.e., $\tilde{c} = max\{\mathbf{c}\}$. It is expected to select the most important features, which are the ones with the highest value, for each feature map. The *max-over-time* pooling operation serves also to make the learned features of a fixed size: it allows to deal with variable sentence lengths and to adopt the learned features in fully connected layers.

This representation is finally used in the *fourth layer*, that is a fully connected softmax layer. It classifies the example into one of the categories of the task. In particular, this layer is characterized by a parameter matrix $\mathbf{S}$ and a bias term $\mathbf{b_c}$ that is used to classify a message, given the learned representations $\tilde{c}$. In particular, the final classification y is obtained through $\text{argmax}_{y \in Y}(softmax(\mathbf{S} \cdot \tilde{c} + \mathbf{b_c}))$, where Y is the set of classes of interest.

In order to reduce the risk of over-fitting, two forms of regularization are applied, as in (Kim, 2014). First, a *dropout* operation over the penultimate layer (Hinton et al., 2012) is adopted to prevent co-adaptation of hidden units by randomly dropping out, i.e., setting to zero, a portion of the hidden units during forward-backpropagation. The second regularization is obtained by constraining the l_2 norm of $\mathbf{S}$ and $\mathbf{b_c}$.

2.1 Injecting Sentiment Information through Polarity Lexicons

In (Kim, 2014), the use of word embeddings is advised to generalize lexical information. These word representations can capture paradigmatic relationships between lexical items. They are best suited to help the generalization of learning algorithms in natural language tasks. However, paradigmatic relationships do not always reflect the relative sentiment between words. In Deep Learning, it is a common practice to make the input representations trainable in the final learning stages. This is a valid strategy, but it makes the learning process more complex. In fact, the number of learnable parameters increases significantly, resulting in the need of more annotated examples in order to adequately estimate them.

We advocate the adoption of a multi-channel input representation, which is typical of CNNs in image processing. A first channel is dedicated to host representations derived from a word embedding. A second channel is introduced to *inject* sentiment information of words through a large-scale polarity lexicon, which is acquired according to the methodology proposed in (Castellucci et al., 2015a). This method leverages on word embedding representations to assign polarity information to words by transferring it from sentences whose polarity is known. The resultant lexicons are called Distributional Polarity Lexicons (DPLs). The process is based on the capability of word embedding to represent both sentences and words in the same space (Landauer and Dumais, 1997). First, sentences (here tweets) are labeled with some polarity classes: in (Castellucci et al., 2015a) this labeling is achieved by apply-

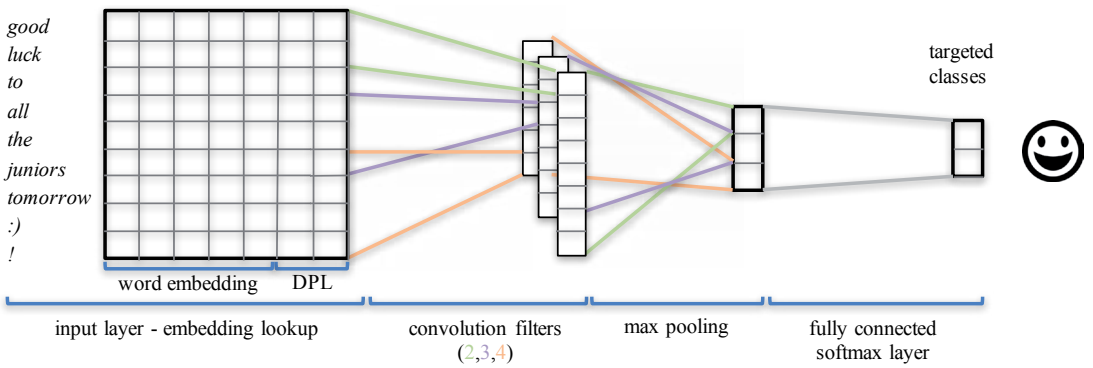


Figure 1: The Convolutional Neural Network architecture adopted for the `Unitor` system.

ing a Distant Supervision (Go et al., 2009) heuristic. The labeled dataset is projected in the embedding space by applying a simple but effective linear combination of the word vectors composing each sentence. Then, a polarity classifier is trained over these sentences in order to emphasize those dimensions of the space more related to the polarity classes. The DPL is generated by classifying each word (represented in the embedding through a vector) with respect to each targeted class, using the confidence level of the classification to derive a word polarity signature. For example, in a DPL the word *ottimo* is 0.89 positive, 0.04 negative and 0.07 neutral (see Table 1). For more details, please refer to (Castellucci et al., 2015a).

Term	w/o DPL	w/ DPL
ottimo (0.89,0.04,0.07)	pessimo eccellente ottima	ottima eccellente fantastico
peggiore (0.17,0.57,0.26)	peggior peggio migliore	peggior peggio peggiori
triste (0.04,0.82,0.14)	deprimente tristissima felice	deprimente tristissima depressa

Table 1: Similar words in the embedding without ($2^{nd}column$) and with ($3^{rd}column$) DPL, whose scores (*positivity, negativity, neutrality*) are in the first column.

This method has two main advantages: first, it allows deriving a signature for each word in the embedding to be used in the CNN; second, this method allows assigning sentiment information to words by observing their usage. This represents an interesting setting to observe sentiment related phenomena, as often a word does not carry a sentiment if not immersed in a context (i.e., a sentence).

As proposed in (Croce et al., 2016), in order to keep limited the computational complexity of the training phase of CNN, we augment each vector from the embedding with the polarity scores derived from the DPL[1]. In Table 1, a comparison of the most similar words of polarity carriers is compared when the polarity lexicon is not adopted (second column) and when the multi-channel schema is adopted (third column). Notice that, the DPL positively affects the vector representations for SA. For example, the word *pessimo* is no longer in set of the 3-most similar words of the word *ottimo*. The polarity information captured in the DPL making words that are semantically related and whose polarity agrees nearer in the space.

2.2 Context-aware model for SA in Twitter

In (Severyn and Moschitti, 2015) a pre-training strategy is suggested for the Sentiment Analysis task. The adoption of heuristically classified tweet messages is advised to initialize the network parameters. The selection of messages is based on the presence of emoticons (Go et al., 2009) that can be related to polarities, e.g. :) and :(. However, selecting messages only with emoticons could potentially introduce many topically unrelated messages that use out-of-domain linguistic expressions and limiting the contribution of the pre-training. We instead suggest to adopt another strategy for the selection of pre-training data. We draw on the work in (Vanzo et al., 2014), where topically related messages of the target domain are selected by considering the *reply-to* or *hashtag contexts* of each message. The former (*conversational context*) is made of the stream of messages belonging to the same conversation in Twitter, while the latter (*hashtag context*) is composed by tweets preceding a target message and sharing at least one hashtag with it. In (Vanzo et al., 2014), these messages are first classified through a

[1] We normalize the embedding and the DPL vectors before the juxtaposition.

context-unaware SVM classifier. Here, we are going to leverage on contextual information for the selection of pre-training material for the CNN. We select the messages both in the *conversation* context, and we classify them with a context-unaware classifier to produce the pre-training dataset.

3 The `Unitor` Classification Workflow

The SENTIPOLC challenge is made of three subtasks aiming at investigating different aspects of the subjectivity of short messages. The first subtask is the Subjectivity Classification that consists in deciding whether a message expresses subjectivity or it is objective. The second task is the Polarity Classification: given a subjective tweet a system should decide whether a tweet is expressing a neutral, positive, negative or conflict position. Finally, the Irony Detection sub-task aims at finding whether a message is expressing ironic content or not. The `Unitor` system tackles each sub-task with a different CNN classifier, resulting in a classification workflow that is summarized in the Algorithm 1: a message is first classified with the *Subjectivity* CNN-based classifier `S`; in the case the message is classified as *subjective* (`subjective=True`), it is also processed with the other two classifiers, the *Polarity* classifier `P` and the *Irony* classifier `I`. In the case the message is first classified as *objective* (`subjective=False`), the remaining classifiers are not invoked.

Algorithm 1 `Unitor` classification workflow.

```
1: function TAG(tweet T, cnn S, cnn P, cnn I)
2:     subjective = S(T)
3:     if subjective==True then
4:         polarity = P(T), irony = I(T)
5:     else
6:         polarity = none, irony = none
7:     end if
       return subjective, polarity, irony
8: end function
```

The same CNN architecture is adopted to implement all the three classifiers and tweets are modeled in the same way for the three sub-tasks. Each classifier has been specialized to the corresponding sub-task by adopting different selection policies of the training material and adapting the output layer of the CNN to the sub-task specific classes. In detail, the *Subjectivity* CNN is trained over the whole training dataset with respect to the classes `subjective` and `objective`. The *Polarity* CNN is trained over the subset of subjective tweets, with respect to the classes `neutral`, `positive`, `negative` and `conflict`. The *Irony* CNN is trained over the subset of subjective tweets, with respect to the classes `ironic` and `not-ironic`.

Each CNN classifier has been trained in the two settings specified in the SENTIPOLC guidelines: *constrained* and *unconstrained*. The constrained setting refers to a system that adopted only the provided training data. For example, in the constrained setting it is forbidden the use of a word embedding generated starting from other tweets. The unconstrained systems, instead, can adopt also other tweets in the training stage. In our work, the constrained CNNs are trained without using a pre-computed word embedding in the input layer. In order to provide input data to the neural network, we randomly initialized the word embedding, adding them to the parameters to be estimated in the training process: in the following, we will refer to the constrained classification workflow as `Unitor`. The unconstrained CNNs are instead initialized with pre-computed word embedding and DPL. Notice that in this setting we do not back-propagate over the input layer. The word embedding is obtained from a corpus downloaded in July 2016 of about 10 millions of tweets. A 250-dimensional embedding is generated according to a Skip-gram model (Mikolov et al., 2013)[2]. Starting from this corpus and the generated embedding, we acquired the DPL according to the methodology described in Section 2.1. The final embedding is obtained by juxtaposing the Skip-gram vectors and the DPL[3], resulting in a 253-dimensional representation for about $290,000$ words, as shown in Figure 1. The resulting classification workflow made of unconstrained classifier is called `Unitor-U1`. Notice that these word representations represent a richer feature set for the CNN, however the cost of obtaining them is negligible, as no manual activity is needed.

As suggested in (Croce et al., 2016), the contextual pre-training (see Section 2.2) is obtained by considering the conversational contexts of the provided training data. This dataset is made of about $2,200$ new messages, that have been classified with the `Unitor-U1` system. This set of

[2]The following settings are adopted: window 5 and min-count 10 with hierarchical softmax

[3]Measures adopting only the Skip-gram vectors have been pursued in the classifier tuning stage; these have highlighted the positive contribution of the DPL.

messages is adopted to initialize the network parameters. In the following, the system adopting the pre-trained CNNs is called `Unitor-U2`.

The CNNs have a number of hyper-parameters that should be fine-tuned. The parameters we investigated are: *size of filters*, i.e., capturing $2/3/4/5$-grams. We combined together multiple filter sizes in the same run. The *number of filters* for each size: we selected this parameter among 50, 100 and 200. The *dropout keep probability* has been selected among 0.5, 0.8 and 1.0. The final parameters has been determined over a development dataset, made of the 20% of the training material. Other parameters have been kept fixed: *batch size* (100), *learning rate* (0.001), *number of epochs* (15) and *L2 regularization* (0.0). The CNNs are implemented in Tensorflow[4] and they have been optimized with the Adam optimizer.

4 Experimental Results

In Tables 2, 3 and 4 the performances of the `Unitor` systems are reported, respectively for the task of Subjectivity Classification, Polarity Classification and Irony Detection. In the first Table (2) the `F-0` measure refers to the F1 measure of the *objective* class, while *F-1* refers to the F1 measure of the *subjective* class. In the Table 3 the `F-0` measure refers to the F1 measure of the *negative* class, while *F-1* refers to the F1 measure of the *positive* class. Notice that in this case, the *neutral* class is mapped to a "not negative" and "not positive" classification and the *conflict* class is mapped to a "negative" and "positive" classification. The *F-0* and *F-1* measures capture also these configurations. In Table 4 the `F-0` measure refers to the F1 measure of the *not ironic* class, while *F-1* refers to the F1 measure of the *ironic* class. Finally, *F-Mean* is the mean between these *F-0* and *F-1* values, and is the score used by the organizers for producing the final ranks.

System	F-0	F-1	F-Mean	Rank
`Unitor-C`	.6733	.7535	.7134	4
`Unitor-U1`	.6784	.8105	.7444	1
`Unitor-U2`	.6723	.7979	.7351	2

Table 2: Subjectivity Classification results

Notice that our unconstrained system (`Unitor-U1`) is the best performing system in recognizing when a message is expressing a subjective position or not, with a final *F-mean* of .7444 (Table 2). Moreover, also the `Unitor-U2` system is capable of adequately classify whether a message is subjective or not. The fact that the pre-trained system is not performing as well as `Unitor-U1`, can be ascribed to the fact that the pre-training material size is actually small. During the classifier tuning phases we adopted also the *hashtag* contexts (about $20,000$ messages) (Vanzo et al., 2014) to pre-train our networks: the measures over the development set indicated that probably the *hashtag* contexts were introducing too many unrelated messages. Moreover, the pre-training material has been classified with the `Unitor-U1` system. It could be the case that the adoption of such added material was not so effective, as instead demonstrated in (Croce et al., 2016). In fact, in that work the pre-training material was classified with a totally different algorithm (Support Vector Machine) and a totally different representation (kernel-based). In this setting, the different algorithm and representation produced a better and substantially different dataset, in terms of covered linguistic phenomena and their relationships with the target classes. Finally, the constrained version of our system, obtained a remarkable score of .7134, demonstrating that the random initialization of the input vectors can be also adopted for the classification of the subjectivity of a message.

System	F-0	F-1	F-Mean	Rank
`Unitor-C`	.6486	.6279	.6382	11
`Unitor-U1`	.6885	.6354	.6620	2
`Unitor-U2`	.6838	.6312	.6575	3

Table 3: Polarity Classification results

In Table 3 the Polarity Classification results are reported. Also in this task, the performances of the unconstrained systems are higher with respect to the constrained one (.662 against .6382). It demonstrates the usefulness of acquiring lexical representations and use them as inputs for the CNNs. Notice that the performances of the `Unitor` classifiers are remarkable, as the two unconstrained systems rank in 2^{nd} and 3^{rd} position. The contribution of the pre-training is not positive, as instead measured in (Croce et al., 2016). Again, we believe that the problem resides in the size and quality of the pre-training dataset.

In Table 4 the Irony Detection results are reported. Our systems do not perform well, as all the submitted systems reported a very low recall

[4] `https://www.tensorflow.org/`

System	F-0	F-1	F-Mean	Rank
Unitor-C	.9358	.016	.4761	10
Unitor-U1	.9373	.008	.4728	11
Unitor-U2	.9372	.025	.4810	9

Table 4: Irony Detection results

for the *ironic* class: for example, the Unitor-U2 recall is only .0013, while its precision is .4286. It can be due mainly to two factors. First, the CNN devoted to the classification of the irony of a message has been trained with a dataset very skewed towards the *not-ironic* class: in the original dataset only 868 over 7409 messages are ironic. Second, a CNN observes local features (bi-grams, tri-grams, ...) without ever considering global constraints. Irony, is not a word-level phenomenon but, instead, it is related to sentence or even social aspects. For example, the best performing system in Irony Detection in SENTIPOLC 2014 (Castellucci et al., 2014) adopted a specific feature, which estimates the violation of paradigmatic coherence of a word with respect to the entire sentence, i.e., a global information about a tweet. This is not accounted for in the CNN here discussed, and ironic sub-phrases are likely to be neglected.

5 Conclusions

The results obtained by the Unitor system at SENTIPOLC 2016 are promising, as the system won the Subjectivity Classification sub-task and placed in $2^n d$ position in the Polarity Classification. While in the Irony Detection the results are not satisfactory, the proposed architecture is straightforward as its setup cost is very low. In fact, the human effort in producing data for the CNNs, i.e., the pre-training material and the acquisition of the Distributional Polarity Lexicon is very limited. In fact, the former can be easily acquired with the Twitter Developer API; the latter is realized through an unsupervised process (Castellucci et al., 2015a). In the future, we need to better model the irony detection problem, as probably the CNN here adopted is not best suited for such task. In fact, irony is a more global linguistic phenomenon than the ones captured by the (local) convolutions operated by a CNN.

References

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. Academia University Press.

Giuseppe Castellucci, Danilo Croce, Diego De Cao, and Roberto Basili. 2014. A multiple kernel approach for twitter sentiment analysis in italian. In *Fourth International Workshop EVALITA 2014*.

Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2015a. Acquiring a large scale polarity lexicon through unsupervised distributional methods. In *Proc. of 20th NLDB*, volume 9103. Springer.

Giuseppe Castellucci, Andrea Vanzo, Danilo Croce, and Roberto Basili. 2015b. Context-aware models for twitter sentiment analysis. *IJCoL vol. 1, n. 1: Emerging Topics at the 1st CLiC-It Conf.*, page 69.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2016. Injecting sentiment information in context-aware convolutional neural networks. *Proceedings of SocialNLP@ IJCAI*, 2016.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*.

Geoffrey Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings EMNLP 2014*, pages 1746–1751, Doha, Qatar, October. Association for Computational Linguistics.

Tom Landauer and Sue Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 86(11), Nov.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proc. of the SIGIR 2015*, pages 959–962, New York, NY, USA. ACM.

Andrea Vanzo, Danilo Croce, and Roberto Basili. 2014. A context-based model for sentiment analysis in twitter. In *Proc. of 25th COLING*, pages 2345–2354.

Tandem LSTM-SVM Approach for Sentiment Analysis

Andrea Cimino and **Felice Dell'Orletta**
Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC–CNR)
ItaliaNLP Lab - *www.italianlp.it*
{andrea.cimino, felice.dellorletta}@ilc.cnr.it

Abstract

English. In this paper we describe our approach to EVALITA 2016 SENTIPOLC task. We participated in all the subtasks with constrained setting: Subjectivity Classification, Polarity Classification and Irony Detection. We developed a tandem architecture where Long Short Term Memory recurrent neural network is used to learn the feature space and to capture temporal dependencies, while the Support Vector Machines is used for classification. SVMs combine the document embedding produced by the LSTM with a wide set of general–purpose features qualifying the lexical and grammatical structure of the text. We achieved the second best accuracy in Subjectivity Classification, the third position in Polarity Classification, the sixth position in Irony Detection.

Italiano. *In questo articolo descriviamo il sistema che abbiamo utilizzato per affrontare i diversi compiti del task SENTIPOLC della conferenza EVALITA 2016. In questa edizione abbiamo partecipato a tutti i sotto compiti nella configurazione vincolata, cioè senza utilizzare risorse annotate a mano diverse rispetto a quelle distribuite dagli organizzatori. Per questa partecipazione abbiamo sviluppato un metodo che combina una rete neurale ricorrente di tipo Long Short Term Memory, utilizzate per apprendere lo spazio delle feature e per catturare dipendenze temporali, e Support Vector Machine per la classificazione. Le SVM combinano la rappresentazione del documento prodotta da LSTM con un ampio insieme di features che descrivono la struttura lessicale e grammaticale del testo. Attraverso questo sistema abbiamo ottenuto la seconda posizione nella classificazione della Soggettività, la terza posizione nella classificazione della Polarità e la sesta nella identificazione dell'Ironia.*

1 Description of the system

We addressed the EVALITA 2016 SENTIPOLC task (Barbieri et al., 2016) as a three-classification problem: two binary classification tasks (Subjectivity Classification and Irony Detection) and a four-class classification task (Polarity Classification).

We implemented a tandem LSTM-SVM classifier operating on morpho-syntactically tagged texts. We used this architecture since similar systems were successfully employed to tackle different classification problems such keyword spotting (Wöllmer et al., 2009) or the automatic estimation of human affect from speech signal (Wöllmer et al., 2010), showing that tandem architectures outperform the performances of the single classifiers.

In this work we used Keras (Chollet, 2016) deep learning framework and LIBSVM (Chang et al., 2001) to generate respectively the LSTM and the SVMs statistical models.

Since our approach relies on morpho-syntactically tagged texts, both training and test data were automatically morpho-syntactically tagged by the POS tagger described in (Dell'Orletta, 2009). In addition, in order to improve the overall accuracy of our system (described in 1.2), we developed sentiment polarity and word embedding lexicons[1] described below.

[1] All the created lexicons are made freely available at the following website: *http://www.italianlp.it/*.

1.1 Lexical resources

1.1.1 Sentiment Polarity Lexicons

Sentiment polarity lexicons provide mappings between a word and its sentiment polarity (positive, negative, neutral). For our experiments, we used a publicly available lexicons for Italian and two English lexicons that we automatically translated. In addition, we adopted an unsupervised method to automatically create a lexicon specific for the Italian twitter language.

Existing Sentiment Polarity Lexicons

We used the Italian sentiment polarity lexicon (hereafter referred to as $OPENER$) (Maks et al., 2013) developed within the OpeNER European project[2]. This is a freely available lexicon for the Italian language[3] and includes 24,000 Italian word entries. It was automatically created using a propagation algorithm and the most frequent words were manually reviewed.

Automatically translated Sentiment Polarity Lexicons

- The Multi–Perspective Question Answering (hereafter referred to as $MPQA$) Subjectivity Lexicon (Wilson et al., 2005). This lexicon consists of approximately 8,200 English words with their associated polarity. In order to use this resource for the Italian language, we translated all the entries through the Yandex translation service[4].

- The Bing Liu Lexicon (hereafter referred to as BL) (Hu et al., 2004). This lexicon includes approximately 6,000 English words with their associated polarity. This resource was automatically translated by the Yandex translation service.

Automatically created Sentiment Polarity Lexicons

We built a corpus of positive and negative tweets following the Mohammad et al. (2013) approach adopted in the Semeval 2013 sentiment polarity detection task. For this purpose we queried the Twitter API with a set of hashtag seeds that indicate positive and negative sentiment polarity. We selected 200 positive word seeds (e.g. "vincere" *to win*, "splendido" *splendid*, "affascinante" *fascinating*), and 200 negative word seeds (e.g., "tradire" *betray*, "morire" *die*). These terms were chosen from the OPENER lexicon. The resulting corpus is made up of 683,811 tweets extracted with positive seeds and 1,079,070 tweets extracted with negative seeds.

The main purpose of this procedure was to assign a polarity score to each n-gram occurring in the corpus. For each n-gram (we considered up to five n-grams) we calculated the corresponding sentiment polarity score with the following scoring function: $score(ng) = PMI(ng, pos) - PMI(ng, neg)$, where PMI stands for pointwise mutual information.

1.1.2 Word Embedding Lexicons

Since the lexical information in tweets can be very sparse, to overcame this problem we built two word embedding lexicons.

For this purpose, we trained two predict models using the word2vec[5] toolkit (Mikolov et al., 2013). As recommended in (Mikolov et al., 2013), we used the CBOW model that learns to predict the word in the middle of a symmetric window based on the sum of the vector representations of the words in the window. For our experiments, we considered a context window of 5 words. These models learn lower-dimensional word embeddings. Embeddings are represented by a set of latent (hidden) variables, and each word is a multidimensional vector that represent a specific instantiation of these variables. We built two Word Embedding Lexicons starting from the following corpora:

- The first lexicon was built using a tokenized version of the itWaC corpus[6]. The itWaC corpus is a 2 billion word corpus constructed from the Web limiting the crawl to the .it domain and using medium-frequency words from the Repubblica corpus and basic Italian vocabulary lists as seeds.

- The second lexicon was built from a tokenized corpus of tweets. This corpus was collected using the Twitter APIs and is made up of 10,700,781 italian tweets.

1.2 The LSTM-SVM tandem system

SVM is an extremely efficient learning algorithm and hardly to outperform, unfortunately these type

[2] http://www.opener-project.eu/

[3] https://github.com/opener-project/public-sentiment-lexicons

[4] http://api.yandex.com/translate/

[5] http://code.google.com/p/word2vec/

[6] http://wacky.sslmit.unibo.it/doku.php?id=corpora

of algorithms capture "sparse" and "discrete" features in document classification tasks, making really hard the detection of relations in sentences, which is often the key factor in detecting the overall sentiment polarity in documents (Tang et al., 2015). On the contrary, Long Short Term Memory (LSTM) networks are a specialization of Recurrent Neural Networks (RNN) which are able to capture long-term dependencies in a sentence. This type of neural network was recently tested on Sentiment Analysis tasks (Tang et al., 2015), (Xu et al., 2016) where it has been proven to outperform classification performance in several sentiment analysis task (Nakov et al., 2016) with respect to commonly used learning algorithms, showing a 3-4 points of improvements. For this work, we implemented a tandem LSTM-SVM to take advantage from the two classification strategies.

Figure 1 shows a graphical representation of the proposed tandem architecture. This architecture is composed of 2 sequential machine learning steps both involved in training and classification phases. In the training phase, the LSTM network is trained considering the training documents and the corresponding gold labels. Once the statistical model of the LSTM neural network is computed, for each document of the training set a document vector (document embedding) is computed exploiting the weights that can be obtained from the penultimate network layer (the layer before the SoftMax classifier) by giving in input the considered document to the LSTM network. The document embeddings are used as features during the training phase of the SVM classifier in conjunction with a set of widely used document classification features. Once the training phase of the SVM classifier is completed the tandem architecture is considered trained. The same stages are involved in the classification phase: for each document that must be classified, an embedding vector is obtained exploiting the previously trained LSTM network. Finally the embedding is used jointly with other document classification features by the SVM classifier which outputs the predicted class.

1.2.1 The LSTM network

In this part, we describe the LSTM model employed in the tandem architecture. The LSTM unit was initially proposed by Hochreiter and Schmidhuber (Hochreiter et al., 1997). LSTM units are

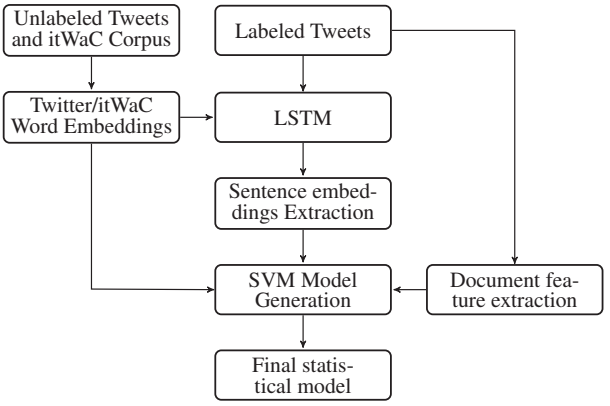


Figure 1: The LSTM-SVM architecture

able to propagate an important feature that came early in the input sequence over a long distance, thus capturing potential long-distance dependencies.

LSTM is a state-of-the-art learning algorithm for semantic composition and allows to compute representation of a document from the representation of its words with multiple abstraction levels. Each word is represented by a low dimensional, continuous and real-valued vector, also known as word embedding and all the word vectors are stacked in a word embedding matrix.

We employed a bidirectional LSTM architecture since these kind of architecture allows to capture long-range dependencies from both directions of a document by constructing bidirectional links in the network (Schuster et al., 1997). In addition, we applied a dropout factor to both input gates and to the recurrent connections in order to prevent overfitting which is a typical issue in neural networks (Galp and Ghahramani , 2015). As suggested in (Galp and Ghahramani , 2015) we have chosen a dropout factor value in the optimum range $[0.3, 0.5]$, more specifically 0.45 for this work. For what concerns the optimization process, categorical cross-entropy is used as a loss function and optimization is performed by the rmsprop optimizer (Tieleman and Hinton, 2012).

Each input word to the LSTM architecture is represented by a 262-dimensional vector which is composed by:

Word embeddings: the concatenation of the two word embeddings extracted by the two available Word Embedding Lexicons (128 dimensions for each word embedding, a total of 256 dimensions), and for each word embedding an extra component was added in order to handle the "unknown word"

(2 dimensions).

Word polarity: the corresponding word polarity obtained by exploiting the Sentiment Polarity Lexicons. This results in 3 components, one for each possible lexicon outcome (negative, neutral, positive) (3 dimensions). We assumed that a word not found in the lexicons has a neutral polarity.

End of Sentence: a component (1 dimension) indicating whether or not the sentence was totally read.

1.2.2 The SVM classifier

The SVM classifier exploits a wide set of features ranging across different levels of linguistic description. With the exception of the *word embedding combination*, these features were already tested in our previous participation at the EVALITA 2014 SENTIPOLC edition (Cimino et al., 2014). The features are organised into three main categories: *raw and lexical text features*, *morpho-syntactic features* and *lexicon features*.

Raw and Lexical Text Features

Topic: the manually annotated class of topic provided by the task organizers for each tweet.

Number of tokens: number of tokens occurring in the analyzed tweet.

Character n-grams: presence or absence of contiguous sequences of characters in the analyzed tweet.

Word n-grams: presence or absence of contiguous sequences of tokens in the analyzed tweet.

Lemma n-grams: presence or absence of contiguous sequences of lemma occurring in the analyzed tweet.

Repetition of n-grams chars: presence or absence of contiguous repetition of characters in the analyzed tweet.

Number of mentions: number of mentions (@) occurring in the analyzed tweet.

Number of hashtags: number of hashtags occurring in the analyzed tweet.

Punctuation: checks whether the analyzed tweet finishes with one of the following punctuation characters: "?", "!".

Morpho-syntactic Features

Coarse grained Part-Of-Speech n-grams: presence or absence of contiguous sequences of coarse–grained PoS, corresponding to the main grammatical categories (noun, verb, adjective).

Fine grained Part-Of-Speech n-grams: presence or absence of contiguous sequences of fine-grained PoS, which represent subdivisions of the coarse-grained tags (e.g. the class of nouns is subdivided into proper vs common nouns, verbs into main verbs, gerund forms, past particles).

Coarse grained Part-Of-Speech distribution: the distribution of nouns, adjectives, adverbs, numbers in the tweet.

Lexicon features

Emoticons: presence or absence of positive or negative emoticons in the analyzed tweet. The lexicon of emoticons was extracted from the site *http://it.wikipedia.org/wiki/Emoticon* and manually classified.

Lemma sentiment polarity n-grams: for each n-gram of lemmas extracted from the analyzed tweet, the feature checks the polarity of each component lemma in the existing sentiment polarity lexicons. Lemma that are not present are marked with the *ABSENT* tag. This is for example the case of the trigram "tutto molto bello" (*all very nice*) that is marked as "*ABSENT-POS-POS*" because *molto* and *bello* are marked as positive in the considered polarity lexicon and *tutto* is absent. The feature is computed for each existing sentiment polarity lexicons.

Polarity modifier: for each lemma in the tweet occurring in the existing sentiment polarity lexicons, the feature checks the presence of adjectives or adverbs in a left context window of size 2. If this is the case, the polarity of the lemma is assigned to the modifier. This is for example the case of the bigram "non interessante" (*not interesting*), where "interessante" is a positive word, and "non" is an adverb. Accordingly, the feature "non_POS" is created. The feature is computed 3 times, checking all the existing sentiment polarity lexicons.

PMI score: for each set of unigrams, bigrams, trigrams, four-grams and five-grams that occur in the analyzed tweet, the feature computes the score given by $\sum_{\text{i}-gram \in tweet} score(\text{i}-gram)$ and returns the minimum and the maximum values of the five values (approximated to the nearest integer).

Distribution of sentiment polarity: this feature computes the percentage of positive, negative and neutral lemmas that occur in the tweet. To overcome the sparsity problem, the percentages are rounded to the nearest multiple of 5. The feature is computed for each existing lexicon.

Most frequent sentiment polarity: the feature returns the most frequent sentiment polarity of the lemmas in the analyzed tweet. The feature is com-

puted for each existing lexicon.

Sentiment polarity in tweet sections: the feature first splits the tweet in three equal sections. For each section the most frequent polarity is computed using the available sentiment polarity lexicons. The purpose of this feature is aimed at identifying change of polarity within the same tweet.

Word embeddings combination: the feature returns the vectors obtained by computing separately the average of the word embeddings of the nouns, adjectives and verbs of the tweet. It computed once for each word embedding lexicon, obtaining a total of 6 vectors for each tweet.

2 Results and Discussion

We tested five different learning configurations of our system: linear and quadratic support vector machines (linear SVM, quadratic SVM) using the features described in section 1.2.2, with the exception of the document embeddings generated by the LSTM; LSTM using the word embeddings described in 1.2.2; A tandem SVM-LSTM combination with linear and quadratic SVM kernels (linear Tandem, quadratic Tandem) using the features described in section 1.2.2 and the document embeddings generated by the LSTM. To test the proposed classification models, we created an internal development set randomly selected from the training set distributed by the task organizers. The resulting development set is composed by the 10% (740 tweets) of the whole training set.

Configuration	Subject.	Polarity	Irony
linear SVM	0.725	0.713	0.636
quadratic SVM	0.740	0.730	0.595
LSTM	0.777	0.747	0.646
linear Tandem	0.764	0.743	0.662
quadratic Tandem	**0.783**	**0.754**	**0.675**

Table 1: Classification results of the different learning models on our development set.

Table 1 reports the overall accuracies achieved by the classifiers on our internal development set for all the tasks. The accuracy is calculated as the F–score obtained using the evaluation tool provided by the organizers. It is worth noting that there are similar trends for what concerns the accuracies of the proposed learning models for all the three tasks. In particular, LSTM outperforms SVM models while the Tandem systems clearly outperform the SVM and LSTM ones. In addition, the quadratic models perform better than the linear ones. These results lead us to choose the linear and quadratic tandem models as the final systems to be used on the official test set.

Configuration	Subject.	Polarity	Irony
best official Runs	**0.718**	0.664	**0.548**
quadratic SVM	0.704	0.646	0.477
linear SVM	0.661	0.631	0.495
LSTM	0.716	**0.674**	0.468
linear Tandem*	0.676	0.650	0.499
quadratic Tandem*	0.713	0.643	0.472

Table 2: Classification results of the different learning models on the official test set.

Table 2 reports the overall accuracies achieved by all our classifier configurations on the official test set, the official submitted runs are starred in the table. The *best official Runs* row reports, for each task, the best official results in EVALITA 2016 SENTIPOLC. As can be seen, the accuracies of different learning models reveal a different trend when tested on the development and the test sets. Differently from what observed in the development experiments, the best system results to be the LSTM one and the gap in terms of accuracy between the linear and quadratic models is lower or does not occur. In addition, the accuracies of all the systems are definitely lower than the ones obtained in our development experiments. In our opinion, such results may depend on the occurrence of out domain tweets in the test set with respect to the ones contained in the training set. Different groups of annotators could be a further motivation for these different results and trends.

3 Conclusion

In this paper, we reported the results of our participation to the EVALITA 2016 SENTIPOLC tasks. By resorting to a tandem LSTM-SVM system we achieved the second place at the Subjectivity Classification task, the third place at the Sentiment Polarity Classification task and the sixth place at the Irony Detection task. This tandem system combines the ability of the bidirectional LSTM to capture long-range dependencies between words from both directions of a tweet with SVMs which are able to exploit document embeddings produced by LSTM in conjunction with a wide set of general-

purpose features qualifying the lexical and grammatical structure of a text. Current direction of research is introducing a character based LSTM (dos Santos and Zadrozny, 2013) in the tandem system. Character based LSTM proven to be particularly suitable when analyzing social media texts (Dhingra et al., 2016).

References

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In *Proceedings of EVALITA '16, Evaluation of NLP and Speech Tools for Italian*. December, Naples, Italy.

Chih-Chung Chang and Chih-Jen Lin. 2001. LIBSVM: a library for support vector machines. *Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm*.

François Chollet. 2016. Keras. *Software available at https://github.com/fchollet/keras/tree/master/keras*.

Andre Cimino, Stefano Cresci, Felice Dell'Orletta, Maurizio Tesconi. 2014. Linguistically-motivated and Lexicon Features for Sentiment Analysis of Italian Tweets. In *Proceedings of EVALITA '14, Evaluation of NLP and Speech Tools for Italian*. December, Pisa, Italy.

Felice Dell'Orletta. 2009. Ensemble system for Part-of-Speech tagging. In *Proceedings of EVALITA '09, Evaluation of NLP and Speech Tools for Italian*. December, Reggio Emilia, Italy.

Bhuwan Dhingra, Zhong Zhou, Dylan Fitzpatrick, Michael Muehl, William Cohen. 2016. Tweet2Vec: Character-Based Distributed Representations for Social Media. In *Proceedings of the 54th Annual Meeting of the ACL*. Berlin, German.

Cìcero Nogueira dos Santos and Bianca Zadrozny. 2014. Learning Character-level Representations for Part-of-Speech Tagging. In *Proc. of the 31st Inter. Conference on Machine Learning (ICML 2014)*.

Yarin Gal and Zoubin Ghahramani. 2015. A theoretically grounded application of dropout in recurrent neural networks. *arXiv preprint arXiv:1512.05287*

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *Neural computation*

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '04*. 368-177, New York, NY, USA. ACM.

Isa Maks, Ruben Izquierdo, Francesca Frontini, Montse Cuadros, Rodrigo Agerri and Piek Vossen. 2014. Generating Polarity Lexicons with WordNet propagation in 5 languages. *9th LREC, Language Resources and Evaluation Conference*. Reykjavik, Iceland.

Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv1:1301.3781.

Saif Mohammad, Svetlana Kiritchenko and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the Seventh international workshop on Semantic Evaluation Exercises, SemEval-2013*. 321-327, Atlanta, Georgia, USA.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani and Veselin Stoyanov. 2016. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681

Duyu Tang, Bing Qin and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of EMNLP 2015*. 1422-1432, Lisbon, Portugal.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-RmsProp: Divide the gradient by a running average of its recent magnitude. In *COURSERA: Neural Networks for Machine Learning.*

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Sara Rosenthal, Veselin Stoyanov and Alan Ritter. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of HLT-EMNLP 2005*. 347-354, Stroudsburg, PA, USA. ACL.

Martin Wöllmer, Florian Eyben, Alex Graves, Björn Schuller and Gerhard Rigoll. 2009. Tandem BLSTM-DBN architecture for keyword spotting with enhanced context modeling *Proc. of NOLISP*.

Martin Wöllmer, Björn Schuller, Florian Eyben and Gerhard Rigoll. 2010. Combining Long Short-Term Memory and Dynamic Bayesian Networks for Incremental Emotion-Sensitive Artificial Listening *IEEE Journal of Selected Topics in Signal Processing*

XingYi Xu, HuiZhi Liang and Timothy Baldwin. 2016. UNIMELB at SemEval-2016 Tasks 4A and 4B: An Ensemble of Neural Networks and a Word2Vec Based Model for Sentiment Classification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*

Lacam&Int@UNIBA at the EVALITA 2016-SENTIPOLC Task

Vito Vincenzo, Covella,
Berardina De Carolis
Department of Computer Science
University of Bari "Aldo Moro", Italy
covinc93@gmail.com,
berardina.decarolis@uniba.it

Stefano Ferilli,
Domenico Redavid
Department of Computer Science
University of Bari "Aldo Moro", Italy
stefano.ferilli@uniba.it,
domenico.redavid@uniba.it

Abstract

English. This paper describes our first experience of participation at the EVALITA challenge. We participated only to the SENTIPOLC Sentiment Polarity subtask and, to this purpose we tested two systems, both developed for a generic Text Categorization task, in the context of the sentiment analysis: SentimentWS and SentiPy. Both were developed according to the same pipeline, but using different feature sets and classification algorithms. The first system does not use any resource specifically developed for the sentiment analysis task. The second one, which had a slightly better performance in the polarity detection subtask, was enriched with an emoticon classifier in order to fit better the purpose of the challenge.

Italiano. *Questo articolo descrive la nostra prima esperienza di partecipazione ad EVALITA. Il nostro team ha partecipato solo al subtask inerente il riconoscimento della Sentiment Polarity, In questo contesot abbiamo testato due sistemi sviluppati genericamente per la Text Categorization applicandoli a questo specifico task: SentimentWS e SentiPy. Entrambi i sistemi usano la stessa pipeline ma con set di feature e algoritmi di classificazione differenti. Il primo sistema non usa alcuna risorsa specifiche per la sentment analysis, mentre il secondo, che si e' classifcato meglio, pur mantendendo la sua genericita' nella classificazione del testo, e' stato arricchito con un classificatore per le emoticon per cercare di renderlo piu' adatto allo scopo della challenge.*

1 Introduction

We tested two systems to analyze the Sentiment Polarity for Italian. They were designed and created to be generic Text Categorization (TC) systems without any specific feature and resource to support Sentiment Analysis. We used them in various domains (movie reviews, opinion about public administration services, mood detection, facebook posts, polarity expressed in the linguistic content of speech interaction, etc.).

Both systems were applied to the EVALITA 2016 SENTIPOLC Sentiment Polarity detection subtask (Barbieri et al., 2016) in order to understand whether, notwithstanding their "general-purpose" and context-independent setting, they were flexible enough to reach a good accuracy. If so, this would mean that the Sentiment Analysis task could be approached without creating special resources for this purpose, which is known to be a costly and critical activity, or that, if available, these resources may improve their performance.

We present here only the results of the constrained runs in which only the provided training data were used to develop the systems.

The first system was entirely developed by the LACAM research group (all the classes used in the pipeline). After studying the effect of different combinations of features and algorithms on the automatic learning of sentiment polarity classifiers in Italian based on the EVALITA SENTIPOLC 2014 dataset, we applied the best one to the training set of EVALITA 2016 in order to participate to the challenge.

The second system was developed using the scikit-learn (Pedregosa et al., 2011) and NLTK libraries (Bird et al., 2009) for building the pipeline and in order to optimize the performance on the provided training set, classifications algorithms and feature sets, different from those used in SentimentWS, were tested.

Even if initially they have been conceived as a generic TC system, with the aim of tuning it

for the SENTIPOLC task, they considered also the emoticons present in the tweets. In the first system this was made by including them in the set of features, while in the second one emoticons were handled by building a classifier whose result was considered to influence the sentiment polarity detection. The results obtained by the two systems are comparable even if the second one shows a better overall accuracy and ranked higher than the first one in the challenge.

2 Systems Description

2.1 SentimentWS

In a previous work (Ferilli et al., 2015) we developed a system for Sentiment Analysis/Opinion Italian. It was called SentimentWS, since it has been initially developed to run as a web-service in the context of opinion coming from web-based applications. SentimentWS casts the Sentiment Classification problem as a TC task, where the categories represent the polarity. To be general and context-independent, it relies on supervised Machine Learning approaches. To learn a classifier, one must first choose what features to consider to describe the documents, and what is the learning method to be exploited. An analysis of the state-of-the-art suggested that no single approach can be considered as the absolute winner, and that different approaches, based on different perspectives, may reach interesting results on different features. As regards the features, for the sake of flexibility, it allows to select different combinations of features to be used for learning the predictive models. As regards the approaches, our proposal is to select a set of approaches that are sufficiently complementary to mutually provide strengths and support weaknesses.

As regards the internal representation of text, most NLP approaches and applications focus on the lexical/grammatical level as a good tradeoff for expressiveness and complexity, effectiveness and efficiency. Accordingly, we have decided to take into account the following kinds of descriptors:

- single normalized words (ignoring dates, numbers and the like), that we believe convey most of informational content in the text;
- abbreviations, acronyms, and colloquial expressions, especially those that are often found in informal texts such as blog posts on the Internet and SMS';
- n-grams (groups of *n* consecutive terms) whose frequency of occurrence in the corpus is above a pre-defined threshold, that sometimes may be particularly meaningful;
- PoS tags, that are intuitively discriminant for subjectivity;
- expressive punctuation (dots, exclamation and question marks), that may be indicative of subjectivity and emotional involvement.

In order to test the system in the context of Sentiment Analysis we added emoticons in the set of features to be considered, due to their direct and explicit relationship to emotions and moods.

As regards NLP pre-processing, we used the TreeTagger (Schmid, 1994) for PoS-tagging and the Snowball suite (Porter, 2001) for stemming. All the selected features are collectively represented in a single vector space based on the real-valued weighting scheme of Term Frequency - Inverse Document Frequency (TF-IDF) (Robertson, 2004). To have values into [0, 1] we use cosine normalization.

To reduce the dimensionality of the vector space, Document Frequency (i.e., removing terms that do not pass a predefined frequency threshold) was used as a good tradeoff between simplicity and effectiveness. To build the classification model we focused on two complementary approaches that have been proved effective in the literature: a similarity-based one (Rocchio) and a probabilistic one (Naive Bayes). SentimentWS combines the above approaches in a committee, where each classifier (i = 1,2) plays the role of a different domain expert that assigns a score sik to category ck for each document to be classified. The final prediction is obtained as class c = arg maxk Sk, considering a function Sk = f (s1k; s2k). There is a wide range of options for function f (Tulyakov et al., 2008). In our case we use a weighted sum, which requires that the values returned by the single approaches are comparable, i.e. they refer to the same scale. In fact, while the Naive Bayes approach returns probability values, Rocchio's classifier returns similarity values, both in [0; 1].

2.2 SentiPy

SentiPy has been developed using the scikit-learn and NLTK libraries for building the pipeline and, in order to optimize the performance on the provided training set, classifica-

tions algorithms and feature sets, different from those used in SentimentWS, were tested. It uses a committee of two classifiers, one for the text component of the message and the other for the emoticons. For the first classifier we use a very simple set of features, any string made at least of two chars, and linear SVC as classification algorithm.

Even if this might seem too simple, we made some experiments in which we tested other configurations of features taking advantage of i) lemmatization, ii) lemmatization followed by POS-tagging, iii) stemming, iv) stemming followed by POS-tagging. All of them were tested with and without removing italian's stopwords (taken from nltk.corpus.stopwords.words("italian")).

We tested also other classification algorithms (Passive Aggressive Classifier, SGDClassifier, Multinomial Naive Bayes), but their performance was less accurate than the one of linear SVC, that we selected.

Before fitting the classifier text preprocessing was performed according to the following steps:

- Twitter's "mentions" (identified by the character '@' followed by the username) and http links are removed;
- retweets special characters ("RT" and "rt") are removed;
- hashtags are "purged", removing the character '#' followed by the string, which is then left unmodified;
- non-BMP utf8 characters (characters outside the Basic Multilingual Plane), usually used to encode special emoticons and emojis used in tweets, are handled by replacing them with their hexadecimal encoding; this is done to avoid errors while reading the files.

After doing the aforementioned experiments using the training and testing sets provided by sentipolc2014, which was also used to fine-tune the parameters used by the LinearSVC algorithm, we compared the most successful approaches: tokenization done using nltk.tokenize.TweetTokenizer followed by stemming and feature extraction simply done by using the default tokenizer provided by *scikit* (it tokenizes the string by extracting words of at least 2 letters).

The best configurations are those shown in Table 1 and Table 2.

Tokenization	Scikit-Learn default tokenizer
Maximum document frequency CountVectorizer parameter	0.5
Maximum number of terms for the vocabulary	unlimited
n-grams	Unigrams and bigrams
Term weights	tf-idf
Vector's normalization	l2
fit_intercept classifier parameter	False
dual classifier parameter	True
Number of iterations over training data	1000
Class balancing	automatic

Table 1: SentiPy - positive vs all best configuration based on Sentipolc 2014.

Tokenization	Scikit-Learn default tokenizer
Maximum document frequency CountVectorizer parameter	0.5
Maximum number of terms for the vocabulary	unlimited
n-grams	Unigrams and bigrams
Term weights	tf-idf
Vector's normalization	l2
fit_intercept classifier parameter	True
dual classifier parameter	True
Number of iterations over training data	1000
Class balancing	automatic

Table 2: SentiPy - negative vs all best configuration based on Sentipolc 2014.

These two configurations, which had the same fine-tuned LinearSVC's parameters, were compared observing the evaluation data obtained testing them on sentipolc2016 training set, taking advantage of a standard common technique: 10-fold cross validation, whose results are shown in Table 3.

The obtained results were comparable therefore we selected the configuration shown in the

first two rows of Table 3 combined with the emoticon classifier since it was not presented in the SentimentWS system.

Configuration	F1-score macro averaging	Accuracy
VotingClassifier default tokenization – *positive vs all*	0,70388	0,77383
VotingClassifier default tokenization – *negative vs all*	0,70162	0,70648
VotingClassifier stemming – *positive vs all*	0,70654	0,75424
VotingClassifier stemming – *negative vs all*	0,6952	0,70351

Table 3: 10-fold on Sentipolc 2016 training set.

As far as the emoticons and emojis are concerned, in this system we decided to exclude them from the features set, solution adopted in SentimentWS, and train a classifier according to the valence with whom the tweet was labeled. This approach may be useful to detect irony or for recognizing valence in particular domains in which emoticons are used with a different meaning. Emoticons and emojis were retrieved using a dictionary of strings and some regular expressions. The emoticons and emojis retrieved are replaced with identifiers, removing all other terms not related to the emoticons, thus obtaining a matrix emoticons-classes. The underlying classifier takes this matrix as input and creates the model that will be used in the classification phase. The algorithm used in the experiments is the Multinomial Naive Bayes.

The committee of classifiers was built using the VotingClassifier class, which is provided by the Scikit-Learn framework. The chosen voting technique is the so called "hard voting": it is based on the majority voting rule; in case of a tie, the classifier will select the class based on the ascending sort order (classifier 1 → class 2; classifier 2 → class 1; class 1 will be the selected class).

3 Experiments

Both systems were tested on other domains before applying them to the SENTIPOLC subtask.
In the results tables, for each class (*positive* and *negative*) 0 represents the value "False" used in the dataset annotations for the specific tweet and class, 1 represents "True", following the task guidelines of Sentipolc 2016. Thus the cell identified by the row *positive* and the column *prec.0* shows the precision related to the tweets with positive polarity annotations set to False. The meaning of the other cells can be obtained analogously.

3.1 SentimentWS Results

SentimentWS was tested initially on a dataset of 2000 reviews in Italian language, concerning 558 movies, taken from http://filmup.leonardo.it/. In this case, classification performance was evaluated on 17 different feature settings using a 5-fold cross-validation procedure. Equal weight was assigned to all classifiers in the SentimentWS committee. Overall accuracy reported in (Ferilli et al., 2015) was always above 81%. When Rocchio outperformed Naive Bayes, accuracy of the committee was greater than that of the components; in the other cases, corresponding to settings that used n-grams, Naive Bayes alone was the winner.

Before tackling the EVALITA 2016 SENTIPOLC task, in order to tune the system on a (hopefully) similar environment, we tested our system on the EVALITA 2014 dataset and determined in this way the combination of features that had a better accuracy on this dataset.

We tested the system using a subset of ~900 tweet (taken from the dataset provided in Sentipolc 2014), in order to find the best configuration of parameters, which resulted to be the following one:

- term normalization: lemmatization;
- minimum number of occurences for a term to be considered: 3
- POS-tags used: NOUN-WH-CLI-ADV-NEG-CON-CHE-DET-NPR-PRE-ART-INTADJ-VER-PRO-AUX
- n-grams: unigrams

With the configuration described above, the system SentimentWS was able to classify the whole test set of Sentipolc 2014 (1935 tweet) obtaining a combined F-score of 0.6285.

The previously mentioned best configuration was also used in one of the two runs sent for Sentipolc 2016, obtaining a combined F-score of 0.6037, as shown in Table 4

class	prec. 0	rec. 0	F-sc. 0	prec. 1	rec. 1	F-sc. 1	F-sc
positive	0.8642	0.7646	0.8113	0.2841	0.4375	0.3445	0.5779
negative	0.7087	0.7455	0.7266	0.5567	0.5104	0.5325	0.6296

Table 4: SentimentWS - Sentipolc 2016 test set - Combined F-score = 0.6037.

3.2 SentiPy Results

With the configuration disccused above SentiPy combined F-score was 0.6281 as shown in Table 5.

class	prec. 0	rec. 0	F-sc. 0	prec. 1	rec. 1	F-sc. 1	F-sc
positive	0.8792	0.7992	0.8373	0.3406	0.4858	0.4005	0.6189
negative	0.7001	0.8577	0.7709	0.6450	0.4130	0.5036	0.6372

Table 5: SentiPy@Sentipolc2016 results (LinearSVC fine-tuned + EmojiCustomClassifier) - Combined F-score = 0.6281.

We made other experiments on the Sentipolc 2016 test set after the deadline of EVALITA. Their results, even if unofficial, show significant improvements, since we managed to get 0.6403 as a combined F-score. We got it by making specific changes in the *positive vs all* classifier: we used lemmatization (without stopwords removal), unigrams (no other n-grams allowed) and the parameter fit_intercept of the LinearSVC algorithm was set to True. The other parameters remained unchanged. No changes have been made to the classifier *negative vs all*.

4 Conclusions

Looking at the results of the Sentiment Polarity detection subtask we were surprised of the overall performance of the systems presented in this paper since they were simply Text Categorization systems. The only integrations to the original systems, in order to tune their performance on the sentiment polarity detection task, concerned emoticons. In SentimentWS these were included in the feature set and SentiPy was enriched with a classifier created for handling emoticons.

Besides the experiments that were executed on the SENTIPOLC dataset, we tested both systems on a dataset of Facebook posts in Italian collected and annotated by a group of researchers in our laboratories. This experiment was important in order to understand whether their performance was comparable to the one obtained in the SENTIPOLC challenge. Results in these cases were encouraging since both systems had a combined F-score higher than 0.8.

We are currently working at the improvement of the performance of the system by tuning it on the Sentiment Analysis context. To this aim we are developing a specific module to handle negation in Italian and, in our future work we plan to integrate the two systems by creating one committee including all the classifiers, moreover we plan to include an approach based on a combination of probabilistic and lexicon (De Carolis et al., 2015).

Reference

Barbieri, Francesco and Basile, Valerio and Croce, Danilo and Nissim, Malvina and Novielli, Nicole and Patti, Viviana 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016). Academia University Press.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: machine learning in python. Journal of Machine Learning Research, 999888, 2825–2830.

Steven Bird, Ewan Klein, and Edward Loper (2009). Natural Language Processing with Python. O'Reilly Media Inc.

Stefano Ferilli , Berardina De Carolis, Floriana Esposito , Domenico Redavid. 2015. Sentiment analysis as a text categorization task: A study on feature and algorithm selection for Italian language. In Proceeding of IEEE International Conference on Data Science and Advanced Analytics (DSAA).

H. Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In Proceedings of International Conference on New Methods in Language Processing, pp. 44–49.

M. F. Porter, 2001. Snowball: A language for stemming algorithms," [Online]. http://snowball.tartarus.org/texts/introduction.html

Stephen Robertson. 2004. Understanding inverse document frequency: On theoretical arguments for IDF. Journal of Documentation.

S. Tulyakov, S. Jaeger, V. Govindaraju, and D. Doermann. 2008. Review of classifier combination methods," ser. Studies in Computational Intelligence (SCI). Springer. vol. 90, pp. 361–386.

B. De Carolis, D. Redavid, A. Bruno. 2015. A Sentiment Polarity Analyser based on a Lexical-Probabilistic Approach . In proceedings of IT@LIA2015 1st AI*IA Workshop on Intelligent Techniques At LIbraries and Archives co-located with XIV Conference of the Italian Association for Artificial Intelligence (AI*IA 2015)

Sentiment Analysis using Convolutional Neural Networks with Multi-Task Training and Distant Supervision on Italian Tweets

Jan Deriu
Zurich University of Applied Sciences
Switzerland
deri@zhaw.ch

Mark Cieliebak
Zurich University of Applied Sciences
Switzerland
ciel@zhaw.ch

Abstract

English. In this paper, we propose a classifier for predicting sentiments of Italian Twitter messages. This work builds upon a deep learning approach where we leverage large amounts of weakly labelled data to train a 2-layer convolutional neural network. To train our network we apply a form of multi-task training. Our system participated in the EvalItalia-2016 competition and outperformed all other approaches on the sentiment analysis task.

In questo articolo, presentiamo un sistema per la classificazione di soggettività e polarità di tweet in lingua italiana. L'approccio descritto si basa su reti neurali. In particolare, utilizziamo un dataset di 300M di tweet per addestrare una convolutional neural network. Il sistema è stato addestrato e valutato sui dati forniti dagli organizzatori di Sentipolc, task di sentiment analysis su Twitter organizzato nell'ambito di Evalita 2016..

1 Introduction

Sentiment analysis is a fundamental problem aiming to give a machine the ability to understand the emotions and opinions expressed in a written text. This is an extremely challenging task due to the complexity and variety of human language.

The sentiment polarity classification task of EvalItalia-2016 [1] (sentipolc) consists of three subtasks which cover different aspects of sentiment detection: $T1$: Subjectivity detection: is the tweet subjective or objective? $T2$: Polarity detection: is the sentiment of the tweet neutral, positive, negative or mixed? $T3$: Irony detection: is the tweet ironic?

The classic approaches to sentiment analysis usually consist of manual feature engineering and applying some sort of classifier on these features (Liu, 2015). Deep neural networks have shown great promises at capturing salient features for these complex tasks (Mikolov et al., 2013b; Severyn and Moschitti, 2015a). Particularly successful for sentiment classification were Convolutional Neural Networks (CNN) (Kim, 2014; Kalchbrenner et al., 2014; Severyn and Moschitti, 2015b; Johnson and Zhang, 2015), on which our work builds upon. These networks typically have a large number of parameters and are especially effective when trained on large amounts of data.

In this work, we use a distant supervision approach to leverage large amounts of data in order to train a 2-layer CNN [2]. More specifically, we train a neural network using the following three-phase procedure: $P1$: creation of word embeddings for the initialization of the first layer based on an unsupervised corpus of 300M Italian tweets; $P2$: distant supervised phase, where the network is pre-trained on a weakly labelled dataset of 40M tweets where the network weights and word embeddings are trained to capture aspects related to sentiment; and $P3$: supervised phase, where the network is trained on the provided supervised training data consisting of 7410 manually labelled tweets.

As the three tasks of EvalItalia-2016 are closely related we apply a form of multitask training as proposed by (Collobert et al., 2011), i.e. we train one CNN for all the tasks simultaneously. This has two advantages: i) we need to train only one model instead of three models, and ii) the CNN has access to more information which benefits the score. The experiments indicate that the multi-task CNN performs better than the single-

[1] http://www.di.unito.it/˜tutreeb/sentipolc-evalita16/index.html

[2] We here refer to a layer as one convolutional and one pooling layer.

task CNN. After a small bugfix regarding the data-preprocessing our system outperforms all the other systems in the sentiment polarity task.

2 Convolutional Neural Networks

We train a 2-layer CNN using 9-fold cross-validation and combine the outputs of the 9 resulting classifiers to increase robustness. The 9 classifiers differ in the data used for the supervised phase since cross-validation creates 9 different training and validation sets.

The architecture of the CNN is shown in Figure 1 and described in detail below.

Sentence model. Each word in the input data is associated to a vector representation, which consists in a d-dimensional vector. A sentence (or tweet) is represented by the concatenation of the representations of its n constituent words. This yields a matrix $\mathbf{S} \in \mathbb{R}^{\mathrm{d}\times\mathrm{n}}$, which is used as input to the convolutional neural network.
The first layer of the network consists of a lookup table where the word embeddings are represented as a matrix $\mathbf{X} \in \mathbb{R}^{\mathrm{d}\times|\mathrm{V}|}$, where V is the vocabulary. Thus the i-th column of X represents the i-th word in the vocabulary V.

Convolutional layer. In this layer, a set of m filters is applied to a sliding window of length h over each sentence. Let $\mathbf{S}_{[i:i+h]}$ denote the concatenation of word vectors $\mathbf{s}_i$ to $\mathbf{s}_{i+h}$. A feature c_i is generated for a given filter $\mathbf{F}$ by:

$$c_i := \sum_{k,j} (\mathbf{S}_{[i:i+h]})_{k,j} \cdot \mathbf{F}_{k,j} \qquad (1)$$

A concatenation of all vectors in a sentence produces a feature vector $\mathbf{c} \in \mathbb{R}^{\mathrm{n}-\mathrm{h}+1}$. The vectors $\mathbf{c}$ are then aggregated over all m filters into a feature map matrix $\mathbf{C} \in \mathbb{R}^{\mathrm{m}\times(\mathrm{n}-\mathrm{h}+1)}$. The filters are learned during the training phase of the neural network using a procedure detailed in the next section.

Max pooling. The output of the convolutional layer is passed through a non-linear activation function, before entering a pooling layer. The latter aggregates vector elements by taking the maximum over a fixed set of non-overlapping intervals. The resulting pooled feature map matrix has the form: $\mathbf{C_{pooled}} \in \mathbb{R}^{m\times\frac{n-h+1}{s}}$, where s is the length of each interval. In the case of overlapping intervals with a stride value s_t, the pooled feature map matrix has the form $\mathbf{C_{pooled}} \in \mathbb{R}^{m\times\frac{n-h+1-s}{s_t}}$. Depending on whether the borders are included or not, the result of the fraction is rounded up or down respectively.

Hidden layer. A fully connected hidden layer computes the transformation $\alpha(\mathbf{W}*\mathbf{x}+\mathbf{b})$, where $\mathbf{W} \in \mathbb{R}^{\mathrm{m}\times\mathrm{m}}$ is the weight matrix, $\mathbf{b} \in \mathbb{R}^{\mathrm{m}}$ the bias, and α the rectified linear ($relu$) activation function (Nair and Hinton, 2010). The output vector of this layer, $\mathbf{x} \in \mathbb{R}^{\mathrm{m}}$, corresponds to the sentence embeddings for each tweet.

Softmax. Finally, the outputs of the hidden layer $\mathbf{x} \in \mathbb{R}^{\mathrm{m}}$ are fully connected to a soft-max regression layer, which returns the class $\hat{y} \in [1, K]$ with largest probability,

$$\hat{y} := \arg\max_j \frac{e^{\mathbf{x}^\intercal \mathbf{w}_j + a_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\intercal \mathbf{w}_k + a_j}}, \qquad (2)$$

where $\mathbf{w}_j$ denotes the weights vector of class j and a_j the bias of class j.

Network parameters. Training the neural network consists in learning the set of parameters $\mathbf{\Theta} = \{\mathbf{X}, \mathbf{F}_1, \mathbf{b}_1, \mathbf{F}_2, \mathbf{b}_2, \mathbf{W}, \mathbf{a}\}$, where $\mathbf{X}$ is the embedding matrix, with each row containing the d-dimensional embedding vector for a specific word; $\mathbf{F}_i, \mathbf{b}_i (i = \{1, 2\})$ the filter weights and biases of the first and second convolutional layers; $\mathbf{W}$ the concatenation of the weights $\mathbf{w}_j$ for every output class in the soft-max layer; and $\mathbf{a}$ the bias of the soft-max layer.

Hyperparameters For both convolutional layers we set the length of the sliding window h to 5, the size of the pooling interval s is set to 3 in both layers, where we use a striding of 2 in the first layer, and the number of filters m is set to 200 in both convolutional layers.

Dropout Dropout is an alternative technique used to reduce overfitting (Srivastava et al., 2014). In each training stage individual nodes are dropped with probability p, the reduced neural net is updated and then the dropped nodes are reinserted. We apply Dropout to the hidden layer and to the input layer using $p = 0.2$ in both cases.

Optimization The network parameters are learned using *AdaDelta* (Zeiler, 2012), which adapts the learning rate for each dimension using only first order information. We used the default hyper-parameters.

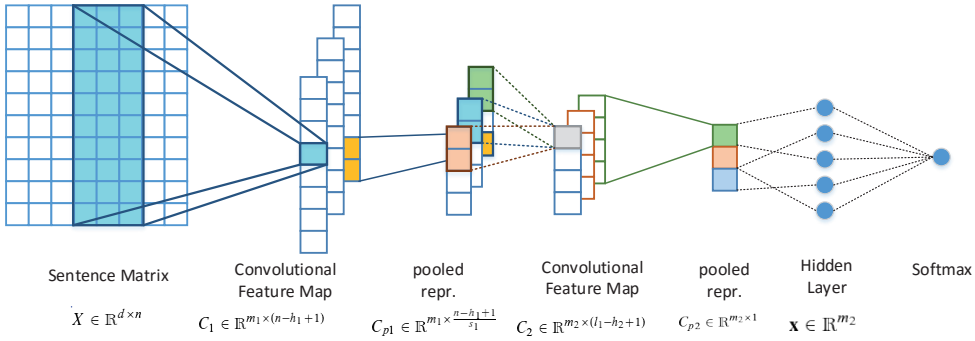


Figure 1: The architecture of the CNN used in our approach.

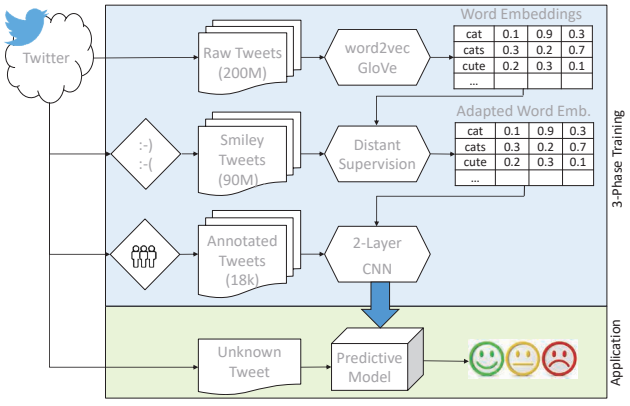


Figure 2: The overall architecture of our 3-phase approach.

3 Training

We train the parameters of the CNN using the three-phase procedure as described in the introduction. Figure 2 depicts the general flow of this procedure.

3.1 Three-Phase Training

Preprocessing We apply standard preprocessing procedures of normalizing URLs, hashtags and usernames, and lowercasing the tweets. The tweets are converted into a list of indices where each index corresponds to the word position in the vocabulary V. This representation is used as input for the lookup table to assemble the sentence matrix S.

Word Embeddings We create the word embeddings in phase $P1$ using word2vec (Mikolov et al., 2013a) and train a skip-gram model on a corpus of 300M unlabelled Italian tweets. The window size for the skip-gram model is 5, the threshold for the minimal word frequency is set to 20 and the number of dimensions is $d = 52$ [3]. The resulting vocabulary contains $890K$ unique words. The word embeddings account for the majority of network parameters ($42.2M$ out of $46.6M$ parameters) and are updated during the next two phases to introduce sentiment specific information into the word embeddings and create a good initialization for the CNN.

Distant Supervised Phase We pre-train the CNN for 1 epoch on an weakly labelled dataset of 40M Italian tweets where each tweet contains an emoticon. The label is inferred by the emoticons inside the tweet, where we ignore tweets with opposite emoticons. This results in 30M positive tweets and 10M negative tweets. Thus, the classifier is trained on a binary classification task.

Supervised Phase During the supervised phase we train the pre-trained CNN with the provided annotated data. The CNN is trained jointly on all tasks of EvalItalia. There are four different binary labels as well as some restrictions which result in 9 possible joint labels (for more details, see Section 3.2). The multi-task classifier is trained to predict the most likely joint-label.
We apply 9-fold cross-validation on the dataset generating 9 equally sized buckets. In each round we train the CNN using early stopping on the held-out set, i.e. we train it as long as the score improves on the held-out set. For the multi-task training we monitor the scores for all 4 subtasks simultaneously and store the best model for each subtask. The training stops if there is no improvement of any of the 4 monitored scores.

Meta Classifier We train the CNN using 9-fold cross-validation, which results in 9 different models. Each model outputs nine real-value numbers

[3] According to the gensim implementation of word2vec using d divisible by 4 speeds up the process.

$\hat{y}$ corresponding to the probabilities for each of the nine classes. To increase the robustness of the system we train a random forest which takes the outputs of the 9 models as its input. The hyper-parameters were found via grid-search to obtain the best overall performance over a development set: Number of trees (100), maximum depth of the forest (3) and the number of features used per random selection (5).

3.2 Data

The supervised training and test data is provided by the EvalItalia-2016 competition. Each tweet contains four labels: $L1$: is the tweet subjective or objective? $L2$: is the tweet positive? $L3$: is the tweet negative? $L4$: is the tweet ironic? Furthermore an objective tweet implies that it is neither positive nor negative as well as not ironic. There are 9 possible combination of labels.
To jointly train the CNN for all three tasks $T1, T2$ and $T3$ we join the labels of each tweet into a single label. In contrast, the single task training trains a single model for each of the four labels separately.
Table 1 shows an overview of the data available.

Table 1: Overview of datasets provided in EvalItalia-2016.

Label	Training Set	Test Set
Total	7410	2000
Subjective	5098	1305
Overall Positive	2051	352
Overall Negative	2983	770
Irony	868	235

3.3 Experiments & Results

We compare the performance of the multi-task CNN with the performance of the single-task CNNs. All the experiments start at the third-phase, i.e. the supervised phase. Since there was no predefined split in training and development set, we generated a development set by sampling 10% uniformly at random from the provided training set. The development set is needed when assessing the generalization power of the CNNs and the meta-classifier. For each task we compute the averaged F1-score (Barbieri et al., 2016). We present the results achieved on the dev-set and the test-set used for the competition. We refer to the set which was held out during a cross validation iteration as fold-set.
In Table 2 we show the average results obtained by the 9 CNNs after the cross validation. The scores show that the CNN is tuned too much towards the held-out folds since the scores of the held-out folds are significantly higher. For example, the average score of the positivity task is 0.733 on the held-out sets but only 0.6694 on the dev-set and 0.6601 on the test-set. Similar differences in sores can be observed for the other tasks as well. To mitigate this problem we apply a random forest on the outputs of the 9 classifiers obtained by cross-validation. The results are shown in Table 3. The meta-classifier outperforms the average scores obtained by the CNNs by up to 2 points on the dev-set. The scores on the test-set show a slightly lower increase in score. Especially the single-task classifier did not benefit from the meta-classifier as the scores on the test set decreased in some cases.
The results show that the multi-task classifier outperforms the single-task classifier in most cases. There is some variation in the magnitude of the difference: the multi-task classifier outperforms the single-task classifier by 0.06 points in the negativity task in the test-set but only by 0.005 points in the subjectivity task.

Set	Task	Subjective	Positive	Negative	Irony
Fold-Set	*Single Task*	0.723	**0.738**	0.721	0.646
	Multi Task	**0.729**	0.733	**0.737**	**0.657**
Dev-Set	*Single Task*	0.696	0.650	0.685	0.563
	Multi Task	**0.710**	**0.669**	**0.699**	**0.595**
Test-Set	*Single Task*	**0.705**	0.652	0.696	0.526
	Multi Task	0.681	**0.660**	**0.700**	**0.540**

Table 2: Average F1-score obtained after applying cross validation.

Set	Task	Subjective	Positive	Negative	Irony
Dev-Set	*Single Task*	0.702	**0.693**	0.695	0.573
	Multi Task	**0.714**	0.686	**0.717**	**0.604**
Test-Set	*Single Task*	0.712	0.650	0.643	0.501
	Multi Task	**0.714**	**0.653**	**0.713**	**0.536**

Table 3: F1-Score obtained by the meta classifier.

4 Conclusion

In this work we presented a deep-learning approach for sentiment analysis. We described the three-phase training approach to guarantee a high quality initialization of the CNN and showed the effects of using a multi-task training approach. To increase the robustness of our system we applied a meta-classifier on top of the CNN. The system was evaluated in the EvalItalia-2016 competition where it achieved 1^{st} place in the polarity task and high positions on the other two subtasks.

References

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *JMLR*, 12:2493–2537.

Rie Johnson and Tong Zhang. 2015. Semi-supervised Convolutional Neural Networks for Text Categorization via Region Embedding. In *NIPS 2015 - Advances in Neural Information Processing Systems 28*, pages 919–927.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *ACL - Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 655–665, Baltimore, Maryland, USA, April.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *EMNLP 2014 - Empirical Methods in Natural Language Processing*, pages 1746–1751, August.

Bing Liu. 2015. *Sentiment Analysis*.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013a. Exploiting Similarities among Languages for Machine Translation. *arXiv*, September.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814.

Aliaksei Severyn and Alessandro Moschitti. 2015a. Twitter Sentiment Analysis with Deep Convolutional Neural Networks. In *38th International ACM SIGIR Conference*, pages 959–962, New York, USA, August. ACM.

Aliaksei Severyn and Alessandro Moschitti. 2015b. UNITN: Training Deep Convolutional Neural Network for Twitter Sentiment Classification. In *SemEval 2015 - Proceedings of the 9th International Workshop on Semantic Evaluation*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Matthew D. Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *arXiv*, page 6.

Tweet2Check evaluation at Evalita Sentipolc 2016

Emanuele Di Rosa
Head of ML and Semantic Analysis
Finsa s.p.a., Via XX Settembre 14
emanuele.dirosa@finsa.it

Alberto Durante
Research Scientist
Finsa s.p.a., Via XX Settembre 14
alberto.durante@finsa.it

Abstract

English. In this paper we present our Tweet2Check tool, provide an analysis of the experimental results obtained by our tool at the Evalita Sentipolc 2016 evaluation, and compare its performance with the state-of-the-art tools that participated to the evaluation. In the experimental analysis, we show that Tweet2Check is: *(i)* the second classified for the irony task, at a distance of just 0.0068 from the first classified; *(ii)* the second classified for the polarity task, considering the unconstrained runs, at a distance of 0.017 from the first tool; *(iii)* in the top 5 tools (out of 13), considering a score that allows to indicate the *most complete-best performing* tools for Sentiment Analysis of tweets, i.e. by summing up the best F-score of each team for the three tasks (subjectivity, polarity and irony); *(iv)* the second best tool, according to the former score, considering together polarity and irony tasks.

Italiano. *In questo paper presentiamo il nostro sistema Tweet2Check, produciamo un'analisi dei risultati sperimentali ottenuti dal nostro strumento nella valutazione effettuata nell'ambito di Evalita Sentipolc 2016, e confrontiamo la sua performance con quella degli altri sistemi partecipanti. Nell'analisi sperimentale, mostriamo che Tweet2Check è:* (i) *il secondo classificato per il task dedicato alla rilevazione dell'ironia, ad una distanza di appena 0.0068 dal primo classificato;* (ii) *il secondo classificato per il task dedicato alla classificazione della polarità, considerando i sistemi* unconstrained*, ad una distanza di 0.017 dal primo classificato;* (iii) *tra i migliori 5 tool (su 13), considerando un punteggio volto ad individuare gli strumenti più completi e meglio performanti per l'analisi del sentiment dei tweet, cioè sommando la migliore F-score di ogni team per i tre task (soggettività, polarità e ironia);* (iv) *il secondo miglior strumento, secondo lo stesso precedente punteggio, considerando insieme i task di polarità e ironia.*

1 Introduction

In this paper we present Tweet2Check, a machine learning-based tool for sentiment analysis of tweets, in which we applied the same approach that we implemented in App2Check and that we have already validated in Di Rosa and Durante (2016-a; 2016-b), showing that it works very well (the most of the times is the best tool) in the field of analysis of apps reviews; moreover, this approach has been also validated on general product/service reviews, since our tool was classified as second at the International Semantic Sentiment Analysis Challenge 2016 (Sack et al., 2016), related to the polarity classification of Amazon product reviews. Our own research interest in participating to the Sentipolc 2016 evaluation is to apply the methodology that was mainly designed to analyze apps reviews, and thus adapted to analyze tweets, and evaluate its performance on tweets. From a research point of view, it is also interesting, to understand if it is possible to obtain good results by applying the same approach to very different domains such as apps reviews and tweets.

Starting from the results provided by the organizers of the Sentipolc 2016 evaluation, we performed an analysis of the results in which we show that Tweet2Check is: *(i)* the second classified for the irony task, at a distance of just 0.0068 from the first classified; *(ii)* the second classified for the polarity task, considering just the unconstrained

runs, at a distance of 0.017 from the first tool; *(iii)* in the top 5 tools (out of 13), considering a score that allows to indicate the *most complete-best performing* tools for Sentiment Analysis of tweets, i.e. by summing up the best F-score of each team for the three tasks (subjectivity, polarity and irony); *(iv)* the second best tool, according to the former score, considering together polarity and irony task.

Finally, we show that Tweet2Check unconstrained runs are overall always better (or almost equal) than the constrained ones. To support our hypothesis, we provide an evaluation of Tweet2Check also on the Sentipolc 2014 (Basile et al., 2014) datasets. This is very important for an industrial tool, since it allows to potentially predict well tweets coming from new domains, by keeping in the training set a higher number of examples discussing different topics, and thus to generalize well from the perspective of the final user.

2 Tweet2Check description

Tweet2Check is an industrial system using an approach in which supervised learning methods are applied in order to build predictive models for the classification of subjectivity, polarity and irony in tweets. The overall machine learning system is an ensemble learning system which combines many different classifiers, each of which is built by us using different machine learning algorithms and implementing different features: this allows to take advantage of different complementary approaches, both discriminative and generative. To this aim, we considered the most well known machine learning algorithms, considering both the most established and the newest approaches. For each task, every classifier has been trained separately; then, the ensemble combines the predictions of the underlying classifiers. The training of the models is performed by considering only the tweets provided by Sentipolc 2016 for the constrained run, and other tweets discussing other topics for the unconstrained run. While performing the training of the models, many features, which are both Twitter-specific and source-independent, are generated. Moreover, some features allowing to ”connect” different tasks are also considered in the pipeline to determine subjectivity, polarity and irony. For example, in the pipeline to determine the polarity of a tweet, a score related to its subjectivity is also included as a feature, thus by reflecting the conceptual connection that there is in reality between subjectivity and polarity: if a tweet can have a polarity assigned is also subjective. The same kind of connection is also applied to the other models.

Tweet2Check does not use just the prediction coming from the predictive model, but it applies also a set of algorithms which takes into account natural language processing techniques, allowing e.g. to also automatically perform topic/named entity extraction, and other resources which have been both handcrafted and automatically extracted. Unfortunately, it is not possible to give more details about the engine due to non-disclosure restrictions.

Tweet2Check is not only constituted by a web service providing access to the sentiment prediction of sentences, but it is also a full user-friendly web application allowing, between other features, to:

- Perform queries on Twitter
- Show the main topics discussed in tweets which are both comment-specific, associated to a specific month or evaluated to the overall results obtained by the query
- Show the polarity, subjectivity and irony associated to each tweet under evaluation
- Show the sentiment of the former extracted topics

A demo of Tweet2Check and its API can be available only for research purposes, by sending a request by email to the first author of the paper. Thus, the results of all of the experiments are repeatable.

3 Experimental Analysis

Considering the Sentipolc 2016 results, we can see that:

- some tools performed very well in one task and very bad in other one (e.g. team2 was the second team for subjectivity and the last one for polarity, team7 was the seventh for subjectivity and the first one for polarity, etc.);
- some other tools show a much better performance on the unconstrained run than on the constrained run (e.g. team1 shows for the subjectivity-unconstrained task a score that is 4% higher than the constrained run).

However, if the goal is to find which are overall the most complete-best performing tools, i.e. performing well considering the contribution that each tool provided on all of the tasks, an overall score/indicator is needed. To this aim, we propose the following score that takes into account, for each team, overall the best run per task. Thus, we introduce formula 1 showing that we consider, given a team and a task, the highest value of F-score between the available runs (considering also constrained and unconstrained runs). Then, in formula 2, we introduce a score per team, calculated as the summation of each contribution provided by each team for the tasks under evaluation (even a subset of them).

$$S_{team,task} = \max_{run}(F_{team,task,run}) \quad (1)$$

$$S_{team} = \sum_{task} S_{team,task} \quad (2)$$

Thanks to this score, it is possible to have an idea of overall the best available tools on: *(i)* each single task; *(ii)* a collection of tasks (couple of tasks at a time in our case), or *(iii)* all of the tasks

Please consider also that this score can be even more restrictive for our tool: we perform better on the unconstrained runs than on the constrained ones, and there are more tools for the constrained runs and performing better than our unconstrained version, so that they would gain positions in the chart (e.g. team3, team4 and team5 for the polarity task perform better on the constrained version). Moreover, we are giving the same equal weight to all of the tasks, even if we focused more on the polarity and irony task which are more related to the original App2Check approach, i.e. more useful and related the evaluation of apps reviews.

Tables 1, 2 and 3 show the results of each single task sorted by the score obtained. The columns contain (from left to right): ranking, team name, the score obtained with formula 1, and a label reporting whether the best run for the team was constrained (c) or unconstrained (u). In Tables 1 and 2 we consider the F-score value coming from the Tweet2Check amended run, representing the correct system answer. For the subjectivity task in Table 1, Tweet2Check does not show good results compared to the other tools, and there is clearly room for further improvements. For all of the other results, Tweet2Check shows good results:

- in Table 2 related to Polarity classification, it is very close to the best result, at a distance of just 0.0188, and it is the second tool considering only the results for the unconstrained run (which are directly comparable)

- in Table 3 related to Irony detection, it is the second best tool, at a distance of just 0.0068 from the first classified.

Tables 4 and 5 show the results obtained using formula 2 considering, respectively, polarity and irony together, and all of the three tasks together[1].

	Team	$\mathbf{S_{team}}$	**con/uncon**
1	team1	0.7444	u
2	team2	0.7184	c
3	team3	0.7134	c
4	team4	0.7107	c
5	team5	0.7105	c
6	team6	0.7086	c
7	team7	0.6937	c/u
8	team8	0.6495	c
9	**Tweet2Check**	**0.6317**	u
10	team10	0.5647	c
11	team11	-	-
12	team12	-	-
13	team13	-	-

Table 1: Subjectivity task at Sentipolc 2016.

In Table 4, Tweet2Check is the second best tool, at a distance of 0.0014 from team4, which is the best tool according to this score. This is clearly our best result at Sentipolc 2016, considering more tasks together, thus highlighting that polarity classification and irony detection are the best tasks performed by Tweet2Check in the current version. In Table 5, we can see that Tweet2Check is the fifth classified, at a distance of 0.0930 from team4, where we consider also the impact of the subjectivity task on the results. In this last case, Tweet2Check is in the top 5 tools chart, over 13 tools. Finally, Tables 6, 7 and 8 report the results obtained training and evaluating Tweet2Check on Evalita Sentipolc 2014 (Basile et al., 2014) datasets. The second and third columns

[1]Since some teams did not participate to all of the tasks, their results are marked as follow:
* The tool did not participate to the Irony task
** The tool participated only to the Polarity task
*** The tool participated only to the Irony task

	Team	S_{team}	con/uncon
1	team7	0.6638	c
2	team1	0.6620	u
3	team4	0.6522	c
4	team3	0.6504	c
5	team5	0.6453	c
6	**Tweet2Check**	**0.6450**	u
7	team10	0.6367	c
8	team11	0.6281	c
9	team12	0.6099	c
10	team6	0.6075	u
11	team8	0.6046	c
12	team2	0.5683	c
13	team13	-	-

Table 2: Polarity task at Sentipolc 2016.

	Team	S_{team}	con/uncon
1	team4	0.5480	c
2	**Tweet2Check**	**0.5412**	c
3	team13	0.5251	c
4	team5	0.5133	c
5	team3	0.4992	c
6	team8	0.4961	c
7	team1	0.4810	u
8	team2	-	-
9	team6	-	-
10	team7	-	-
11	team10	-	-
12	team11	-	-
13	team12	-	-

Table 3: Irony task at Sentipolc 2016.

of the these tables contain, respectively, the F-score of the constrained and the unconstrained runs (in bold the best results). We can see in Table 6 that Tweet2Check ranks first for subjectivity in the unconstrained run, and second for the constrained run. In Tables 7 and 8 Tweet2Check is the best tool for both polarity and irony. Moreover, since we think that Tweet2Check is always better on the unconstrained settings, we decided to further experimentally confirm this observation, and we trained Tweet2Check on the training set of Sentipolc 2014 with the same approach we used for the 2016 edition; thus, we tested it on the test set of the former Sentipolc 2014 evaluation. We show that, also in this case, Tweet2Check unconstrained runs perform better than the constrained

	Team	S_{team}
1	team4	1.2002
2	**Tweet2Check**	**1.1862**
3	team5	1.1586
4	team3	1.1496
5	team1	1.1430
6	team8	1.1007
7	team7*	0.6638
8	team10*	0.6367
9	team11**	0.6281
10	team12**	0.6099
11	team6*	0.6075
12	team2*	0.5683
13	team13***	0.5251

Table 4: The best performing tools on the Polarity and Irony tasks.

	Team	S_{team}
1	team4	1.9109
2	team1	1.8874
3	team5	1.8691
4	team3	1.8630
5	**Tweet2Check**	**1.8179**
6	team8	1.7502
7	team7*	1.3575
8	team6*	1.3161
9	team2*	1.2867
10	team10*	1.2014
11	team11**	0.6281
12	team12**	0.6099
13	team13***	0.5251

Table 5: The best performing tools on the three tasks.

ones, and that our tool is the best tool compared to the tools that participated in 2014.

4 Conclusion

In this paper we presented Tweet2Check and discussed the analysis of the results from Sentipolc 2016, showing that our tool is: *(i)* the second classified for the irony task, at a distance of just 0.0068 from the first classified; *(ii)* the second classified for the polarity task, considering the unconstrained runs, at a distance of 0.017 from the first tool; *(iii)* in the top 5 tools (out of 13), considering a score that allows to indicate the *most complete-best performing* tools for Sentiment Analysis of tweets, i.e. by summing up the best F-score of

Team	F(C)	F(U)
uniba2930	**0.7140**	0.6892
Tweet2Check	0.6927	**0.6903**
UNITOR	0.6871	0.6897
IRADABE	0.6706	0.6464
UPFtaln	0.6497	-
ficlit+cs@unibo	0.5972	-
mind	0.5901	-
SVMSLU	0.5825	-
fbkshelldkm	0.5593	-
itagetaruns	0.5224	-

Table 6: Tweet2Check ranking on the Sentipolc 2014 subjectivity task.

Team	F(C)	F(U)
Tweet2Check	**0.7048**	**0.7142**
uniba2930	0.6771	0.6638
IRADABE	0.6347	0.6108
CoLingLab	0.6312	-
UNITOR	0.6299	0.6546
UPFtaln	0.6049	-
SVMSLU	0.6026	-
ficlit+cs@unibo	0.5980	-
fbkshelldkm	0.5626	-
mind	0.5342	-
itagetaruns	0.5181	-
Itanlp-wafi*	0.5086	-
*amended run	0.6637	-

Table 7: Tweet2Check ranking on the Sentipolc 2014 polarity task.

each team for the three tasks (subjectivity, polarity and irony); *(iv)* the second best tool, according to the former score, considering together polarity and irony tasks.

Team	F(C)	F(U)
Tweet2Check	**0.5915**	-
UNITOR	0.5759	**0.5959**
IRADABE	0.5415	0.5513
SVMSLU	0.5394	-
itagetaruns	0.4929	-
mind	0.4771	-
fbkshelldkm	0.4707	-
UPFtaln	0.4687	-

Table 8: Tweet2Check ranking on the Sentipolc 2014 irony task.

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Association for Computing Machinery. 1983. *Computing Reviews*, 24(11):503–512.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Frencesco Barbieri and Valerio Basile and Danilo Croce and Malvina Nissim and Nicole Novielli and Viviana Patti. 2016. *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press

Emanuele Di Rosa and Alberto Durante LREC 2016 2016. *App2Check: a Machine Learning-based system for Sentiment Analysis of App Reviews in Italian Language* in Proc. of the 2nd International Workshop on Social Media World Sensors, pp. 8-11. http://ceur-ws.org/Vol-1696/

Emanuele Di Rosa, Alberto Durante. *App2Check extension for Sentiment Analysis of Amazon Products Reviews*. In Semantic Web Challenges Vol. 641-1, CCIS Springer 2016

Diego Reforgiato. Results of the Semantic Sentiment Analysis 2016 International Challenge https://github.com/diegoref/SSA2016

ESWC 2016 Challenges http://2016.eswc-conferences.org/program/eswc-challenges

Harald Sack, Stefan Dietze, Anna Tordai. *Semantic Web Challenges*. 2016. CCIS Springer 2016. Third SemWebEval Challenge at ESWC 2016.

Valerio Basile and Andrea Bolioli and Malvina Nissim and Viviana Patti and Paolo Rosso. *Overview of the Evalita 2014 SENTIment POLarity Classification Task*. 2014.

Filipe N. Ribeiro, Matheus Araújo, Pollyanna Gonçalves, Marcos André Gonçalves, Fabrício Benevenuto. *SentiBench - a benchmark comparison of state-of-the-practice sentiment analysis methods* - In EPJ Data Science 2016. 2014.

Computational rule-based model for Irony Detection in Italian Tweets

Simona Frenda
FICLIT - University of Bologna, Italy
simona.frenda@gmail.com

Abstract

English. In the domain of Natural Language Processing (NLP), the interest in figurative language is enhanced, especially in the last few years, thanks to the amount of linguistic data provided by web and social networks. Figurative language provides a non-literary sense to the words, thus the utterances require several interpretations disclosing the play of signification. In order to individuate different meaning levels in case of ironic texts detection, it is necessary a computational model appropriated to the complexity of rhetorical artifice. In this paper we describe our rule-based system of irony detection as it has been presented to the SENTIPOLC task of EVALITA 2016, where we ranked third on twelve participants.

Italiano. *Nell'ambito del Natural Language Processing (NLP) l'interesse per il linguaggio figurativo è particolarmente aumentato negli ultimi anni, grazie alla quantità d'informazione linguistica messa a disposizione dal web e dai social network. Il linguaggio figurativo conferisce alle parole un senso che va oltre quello letterale, pertanto gli enunciati richiedono interpretazioni plurivoche che possano svelare i giochi di significato del discorso. Nel caso specifico del riconoscimento automatico di un testo ironico, infatti, determinare la presenza di diversi gradi di significazione esige un modello computazionale adeguato alla complessità dell'artificio retorico. In questo articolo descriviamo il nostro sistema "rule-based" dedito al riconoscimento dell'ironia che ha partecipato al task SENTIPOLC di EVALITA 2016, nel quale ci siamo classificati terzi su dodici partecipanti.*

1 Introduction

The amount of texts available on the web and especially in social networks has become a source of linguistic information especially for the Sentiment Analysis. For instance, on Twitter, where the length of tweets is limited (140 characters), users are encouraged to use some creative devices in order to communicate their opinions. In particular they express their emotions or feelings through some morphosyntactic elements or conventional expedients, such as: emoticons, hashtags, heavy punctuation, etc. It seems that these elements represent a substitution of typical gestures and tones of oral communication. In this research we used some linguistic features, frequently found in ironic tweets, as referent points to create the rules of our irony detection system in Italian tweets.

The results we gained are promising and reveal the features considered can be good ironic clues to identify ironic texts.

In the following section we synthetically describe the state of art about irony detection. In the third and fourth sections we present our approach, describing the linguistic resources used and data processing. The fifth section contains the description of linguistic features, and finally in the sixth section we present the results obtained in SENTIPOLC evaluation.

2 Related Work

Although the difficulties of research, it is evident in the literature an attempt to understand this linguistic phenomenon and develop some computational models to detect or generate irony.

In the 90s Lessard and Levison (1992, 1993)[1] and Binsted and Ritchie (1994, 1997)[2] developed the first joke generators and recently Stock and Strapparava (2006) realized HAHAcronym, a system designed to generate and re-analyze the acronyms, considering semantic opposition and rhythm criteria.

The research described by Utsumi (1996) was one of the first approaches to automatic irony processing, even though it was too abstract for a computational framework. In 2009, Veale and Hao noted that English figurative comparisons

1Ritchie (2009: 73).
2Ritchie (2009: 73).

(as X as Y) are often used to express ironic opinions, especially when the marker "about" is present (about as X as Y). Recently, Reyes et al. (2013) produced a multidimensional model for detecting irony on Twitter based on four conceptual features: signatures (pointedness, counterfactuality, and temporal compression), unexpectedness (temporal imbalance and contextual imbalance), style and emotional scenarios (activation, imagery, and pleasantness described by Whissel, 2009[3]). Barbieri and Saggion (2014) proposed a model based on a group of seven sets of lexical and semantic features of the words in a tweet: frequency, written-spoken style, intensity of adverbs and adjectives, structure (punctuation, length, emoticons), sentiments, synonyms and ambiguity.

Karoui et al. (2015) focused on the presence of negation markers as well as on both implicit and explicit opposition in French ironic tweets. Moreover, this research highlights the importance of surface traits in ironic texts, such as: punctuation marks (González-Ibáñez et al., 2011), sequence or combination of exclamation and question marks (Carvalho et al., 2009; Buschmeier et al., 2014), tweet length (Davidov et al., 2010), interjections (González-Ibáñez et al., 2011), words in capital letters (Reyes et al., 2013), emoticons (Buschmeier et al., 2014), quotations (Tsur et al., 2010)[4], slang words (Burfoot and Baldwin, 2009)[5] and opposition words, as "but" or "although" (Utsumi, 2004)[6].

Carvalho et al. (2009) distinguished eight "clues" for irony detection in some comments (each consisting of about four sentences) from a Portuguese online newspaper. Their attention focused on positive comments because in a previous research they showed that positive sentences are more subjected to irony and it is more difficult to recognize their true polarity. So the idea is to identify the irony in apparently positive sentences that require the presence of at least one positive adjective or noun in a window of four words. Carvalho et al. (2009) based their model on both oral and gestural "clues" of irony, such as: emoticons, heavy punctuation, quotation marks, onomatopoeic expressions for laughter and positive interjections and, on the other hand, on specific morphosyntactic constructions, such as: the diminutive form of NE, the demonstrative determiners before NE, the pronoun "*tu*" specifically referred or embedded in the morphology of the verb "*ser*".

Our work proposes an adaptation for some of these clues, increased by other surface features, to Italian irony detection in Twitter.

3 Methodology

Approaching the detection of irony in tweets means to understand how people, especially net users, make irony. We try to approach this hard work by analyzing the corpus of tweets and identifying possible ironic clues. Once identified, surface features common to ironic tweets are inserted as binary rules in our system.

Our rule-based system, written in Perl, finds ironic features (described in section 5) in tweets and consequently distinguishes the ironic ones from the non-ironic.

In the following sections we describe resources used, data processing, ironic clues and the results obtained in the EVALITA 2016 SENTIPOLC task.

4 Analysis of corpus

For this research we used a corpus of tweets provided by SENTIPOLC organizers (Barbieri et al., 2016). This training set is composed of 7410 tweets labeled according to the criteria of subjectivity, overall and literal polarity (positive/neutral/negative/mixed), irony and political topic.

4.1 Resources

For the analysis and processing of Italian tweets we used some linguistic resources available online, such as:

- *Sentiment Lexicon* LOD (*Linked Open Data*). Developed by the Institute for Computational Linguistics "A. Zampolli", it contains 24.293 lexical entries annotated with positive/negative/neutral polarity.
- Morph-it! (Zanchetta and Baroni, 2005). It is a lexicon of inflected forms of 34.968 lemma (extracted from the corpus of "La Repubblica") with their morphological features.

A tweet is composed of different essential elements for linguistic analysis, as interjections and emoticons. We therefore developed a lexicon of interjections and a list of emoticons described summarily below:

3Reyes et al. (2013: 249).
4Karoui et al. (2015).
5Karoui et al. (2015).
6Karoui et al. (2015).

- The interjections, extracted from Morph-it! and Treccani[7], are manually annotated with their polarity. The annotation has been developed with the support of Vocabolario Treccani, while the sentiment lexicon has been used to label improper interjections (see Table 1).
- The emoticons, extracted from Wikipedia, are subdivided in EMOPOS, EMONEG and EMOIRO, according to the classification of Di Gennaro et al. (2014) and Wikipedia description[8], especially for the ironic annotation (see Table 2).

Positive	Negative	Neutral
evviva	mah	boh
urrà	macché	mhm
complimenti	bah	chissà
congratulazioni	puah	beh

Table 1: Example of annotated lexicon of interjections.

Label	Emoticon
EMOPOS	=) =] :D (-: [-: (-; [-; :-> :) :-) (; ;)
EMONEG	:[=(:-(:'(:-/ :/ :-> :\> :/ =/ =\ :L =L :S
EMOIRO	^^ ^.^ :P xP ^3^ ^L^ ^_^ ^-^ ^w^

Table 2: Example of annotated list of emoticons.

4.2 Data Processing

Incoming file processed by our system has been previously lemmatized and syntactically annotated by TreeTagger (Schmid, 1994) with Italian tagset provided by Baroni.

Nevertheless, before syntactic analysis, we applied the rules of substitution and elimination of some textual elements, in order to clean up the texts and avoid hampering the process of POStagging and lemmatization of TreeTagger. In particular:

- the label EMOPOS replaces positive emoticons;
- the label EMONEG replaces negative emoticons;
- the label EMOIRO replaces ironic emoticons;
- the characters of url are removed.

This method allows us to clean up the texts from those characters that may hinder the analysis of data and ironic clues retrieval.

5 Features

In section 2 we have presented the research of Carvalho et al. (2009) which demonstrated how the most productive patterns (with a precision from 45% to 85%) are the ones related to orality and gesture, as emoticons or expressions for laughter. Based on this analysis, we try to recognize ironic tweets with a system designed to find ironic clues into the texts. Some of these clues are adapted to Italian language from Portuguese, while some other features are individuated during the analysis of the tweets.

All of these features are used as binary rules in our system to classify the texts in ironic and non-ironic.

5.1 Positive Interjections

Ameka (1992)[9] describes the interjections as "relatively conventionalized vocal gestures which express a speaker's mental state, action or attitude or reaction to a situation". These linguistic elements are used as simple ways to communicate user's feelings or moods.

In previous researches interjections were represented as good humor clues. Kreuz and Caucci (2007) tried to determine if specific lexical factors might suggest the interpretation of a statement as sarcastic. They demonstrated with a test that the presence of interjections is a good predictor for the readers. They provided a group of students with some extracts from various works, a part of which originally contained the word "sarcastically". Students were able to classify correctly the extracts where the word "sarcastically" was deleted thanks to the interjections.

Carvalho et al. (2009) noted that positive interjections has very often an ironical use in apparently positive utterances.

Taking into consideration these precedent researches, we consider improper and proper interjections annotated with positive polarity (see Table 1 in section 4.1). Improper interjections are

7http://www.treccani.it

8Wikipedia version of the 6th of June.

9Lindbladh (2015: 1).

usually followed by exclamations or question marks, which suggest a rising intonation ("*sicuro!*"), whereas proper ones (or onomatopoeic expressions) are sometimes added to the phrase without any punctuation characters ("*ah dimenticavo*", "*ah comunque*").

5.2 Expressions with "*che*"

The adjective or pronoun "*che*" can be used with exclamatory intention in expressions such as "*che ridere*", "*che educato*", "*che sorpresa*". Like interjections, these expressions are used as marks to express user's emotions and their ironic intent.

5.3 Pronoun "*tu*" and Verb Morphology

The use of pronoun "*tu*" and its morphological inflection of the verb "*essere*" expresses a high degree of proximity between the user and the person it refers to (Carvalho et al., 2009). For instance, if this person is a popular politician, this degree of familiarity is fake or artificial and it is usually used ironically in the tweets.

5.4 Disjunctive Conjunction

In the training set we note how disjunctive conjunctions ("*o*", "*oppure*") are used to introduce an alternative between two propositions or concepts which may belong to very different semantic domains (for example: *In televisione stamattina: i cartoni animati o Mario Monti.[…]*). This strange combination of ideas surprises the readers and suggests them a possible ironic interpretation of the message.

5.5 Onomatopoeic Expressions for laughter

Onomatopoeic expressions for laughter (the most diffused are "*ahah*", "*hehe*" and "*ihih*") are usually used in humorous texts (Carvalho et al., 2009; Buschmeier et al., 2014) with their variants (in capital letters or with repetitions). They represent some marks which inform the reader about the user's mood and also suggest that the tweet must be interpreted in a figurative sense.

5.6 Ironic Emoticons

Users utilize emoticons to show their facial expressions as well as their emotions in the texts. Tavosanis (2010) presents a macro-classification of emoticons: expressive, decorative/pleasant and of morphosyntactic substitution, which stand for a word or a whole phrase.

In our research we only consider expressive emoticons which add information about the user's mood. In particular we focus on the ironic emoticons, those which express joking or ironic intention (see section 4.1). We have distinguished EMOIRO from EMOPOS because positive emoticons (considered in Carvalho et al., 2009 and González-Ibáñez et al., 2011) are frequently used to express a humorous intention, not specifically ironic.

5.7 Hashtag

Hashtag is a special element in the syntax of tweets used to connect those ones containing the same keywords (which may be a part of the speech) or phrases as *#mobbastaveramenteperò*.

The user communicates through hashtags several information about events, people they refers to and the topic of message. We focus on hashtags that may suggest to the readers an ironic connotation of the message as #lol and #ironia, and on others that we extracted from ironic tweets in the training set: #stranezze, #Ahahahahah, #benecosì, etc.

5.8 Regional Expressions

It seems that regional expressions are utilized by users in ironic texts to underline their own mood and emotions. In particular, common constructions deriving from local use may be: "*annamo bene*", "*namo bene*" and "*ce*" followed by the verb (e.g. "*ce vuole*", "*ce sta*", "*ce potrebbe*"), as in this ironic tweet: "*@zdizoro t'appassionerà sapè che nel prossimo governo #Monti ce potrebbe rimanè MaryStar Gelmini, come n'incrostazione*".

5.9 Quotation Marks

We focus on the use of quotation marks as a sign for the readers to interpret non-literally the content of text. In fact, in the social networks these elements are frequently used to underline the possible different meanings of the word between quotation marks, and emphasize the ironic content.

5.10 Heavy Punctuation

In web communication the punctuation plays an important role in the expression of the emotions and feelings. Several researches (González-Ibáñez et al., 2011; Kreuz and Caucci, 2007; Carvalho et al., 2009a; Buschmeier et al., 2014; Davidov et al. 2010; Karoui et al., 2014) considered the punctuation as a surface feature to signal humorous texts. In particular we focus on combi-

nation of question and exclamation marks to irony detection.

6 Results

Our system is evaluated on the SENTIPOLC official test data composed of 3000 tweets and the values of precision, recall and average F-score are calculated using the evaluation tool provided by the organizers (Barbieri et al., 2016). As we can see from Table 3, official results of our system are promising, although our research in this domain has to be improved.

Rank	F-score
1	0.548
2	0.5412
3	**0.5251**
4	0.5162
5	0.5133
6	0.4992
7	0.4961
8	0.4872
9	0.481
10	0.4761
11	0.4728
12	0.4725

Table 3: Official results and ranking of Irony Detection sub-task.

7 Conclusion

In this paper we have described our computational model based on linguistic features which have proven to be good clues for the identification of ironic texts. Nonetheless, in future works we plan to examine in depth semantic inconsistencies and ambiguities, amusing wordplay and rhymes that may surprise the reader. In conclusion, we think that a good detection of irony is possible if all the levels of linguistic analysis are considered.

References

Francesco Barbieri and Horacio Saggion. 2014. Modelling Irony in Twitter, Features Analysis and Evaluation. *Language Resources and Evaluation conference, LREC.* Reykjavik, Iceland.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTIment POLarity Classification Task. *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. Academia University Press.

Konstantin Buschmeier, Philipp Cimiano, and Roman Klinger. 2014. An impact analysis of features in a classification approach to irony detection in product reviews. *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Baltimore, Maryland, USA. 42–49.

Paula Carvalho, Luís Sarmento, Mário J. Silva and Eugénio De Oliveira. 2009. Clues for detecting irony in user-generated contents: Oh...!! it's "so easy" ;-). *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*. ACM. 53–56.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10. Stroudsburg, PA, USA. Association for Computational Linguistics. 107–116.

Pierluigi Di Gennaro, Arianna Rossi and Fabio Tamburini. 2014. The FICLIT+CS@UniBO System at the EVALITA 2014 Sentiment Polarity Classification Task. *Proceedings of the Fourth International Workshop EVALITA 2014*. Pisa University Press.

Roberto González-Ibáñez, Smaranda Muresan and Nina Wacholder. 2011. Identifying Sarcasm in Twitter: A Closer Look. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: shortpapers*. Portland, Oregon. 581–586.

Jihen Karoui, Farah Benamara Zitoune, Veronique Moriceau, Nathalie Aussenac-Gilles and Lamia Hadrich Belguith. 2015. Detection automatique de l'ironie dans les tweet en francais. *22eme Traitement Automatique des Langues Naturelles.* Caen.

Roger J. Kreuz and Gina M. Caucci. 2007. Lexical Influences on the Perception of Sarcasm. *Proceedings of the Workshop on Computational Approaches to Figurative Language*. Rochester, NY. 1–4.

Sara Lindbladh. 2015. La semantica e pragmatica dei segnali discorsivi italiani – un confronto tra bene, va bene, be' e va be'. *Seminarium 27 oktober.* Universita di Uppsala, Sweden.

Antonio Reyes, Paolo Rosso and Tony Veale. 2013. A multidimensional approach for detecting irony in-

Twitter. *Language Resources and Evaluation*. 47: 239–268.

Graeme Ritchie. 2009. Can computers create humor? *AI Magazine*. Volume 30, No. 3. 71-81.

Helmut Schmid. 1994. Probabilistic Part-of-Speech-Tagging Using Decision Trees. *Proceedings of International Conference on New Methods in Language Processing*, Manchester, UK.

Oliviero Stock and Carlo Strapparava. 2006. Laughing with HAHAcronym, a computational humor system. *Proceedings of the Twenty-First National Conference on Artificial Intelligence* (AAAI-06), Boston, Massachusetts.

Mirko Tavosanis. 2010. *L'italiano del web.* Carocci. Roma.

Akira Utsumi. 1996. A unified theory of irony and its computational formalization. *Proceedings of the 16th conference on computational linguistics*. Association for Computational Linguistics. Morristown, NJ. 962–967.

Tony Veale and Yanfen Hao. 2009. Support structures for linguistic creativity: A computational analysis of creative irony in similes. *Proceedings of CogSci 2009, the 31st annual meeting of the cognitive science society*. 1376–1381.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! A free corpus-based morphological resource for the Italian language. *Proceedings of Corpus Linguistics 2005*. University of Birmingham, Birmingham, UK.

On the performance of B4MSA on SENTIPOLC'16

Daniela Moctezuma
CONACyT-CentroGEO
Circuito Tecnopolo Norte No. 117,
Col. Tecnopolo Pocitos II, C.P. 20313, Ags, México
dmoctezuma@centrogeo.edu.mx

Eric S. Tellez
Mario Graff
Sabino Miranda-Jiménez
CONACyT-INFOTEC
Circuito Tecnopolo Sur
No 112, Fracc. Tecnopolo Pocitos II,
Ags, 20313, México.
eric.tellez@infotec.mx
mario.graff@infotec.mx
sabino.miranda@infotec.mx

Abstract

This document describes the participation of the INGEOTEC team in SENTIPOLC 2016 contest. In this participation two approaches are presented, B4MSA and B4MSA + EvoDAG, tested in Task 1: Subjectivity classification and Task 2: Polarity classification. In case of polarity classification, one constrained and unconstrained runs were conducted. In subjectivity classification only a constrained run was done. In our methodology we explored a set of techniques as lemmatization, stemming, entity removal, character-based q-grams, word-based n-grams, among others, to prepare different text representations, in this case, applied to the Italian language. The results show the official competition measures and other well-known performance measures such as macro and micro F1 scores.

Italiano. *Questo documento descrive la partecipazione del team INGEOTEC alla competizione SENTIPOLC 2016. In questo contributo sono presentati due approcci, B4MSA e B4MSA + EvoDAG, applicati al Task 1: Subjectivity classification e Task 2: Polarity classification. Nel caso della classificazione della polarit, sono stati sottomessi un run constrained ed un run unconstrained. Per la classificazione della soggettivita, stato sottomesso solo un run constrained. La nostra metodologia esplora un insieme di tecniche come lemmatizzazione, stemming, rimozione di entit, q-grammi di caratteri, n-grammi di parole, ed altri, al fine di ottenere diverse rappresentazioni del testo. In questo caso essa applicata alla lingua italiana. I risultati qui presentati sono due: le metriche della competizione ufficiale ed altre misure note della performance, come macro F1 e micro F1.*

1 Introduction

Nowadays, the sentiment analysis task has become a problem of interest for governments, companies, and institutions due to the possibility of sensing massively the mood of the people using social networks in order to take advantage in decision-making process. This new way to know *what are people thinking* about something imposes challenges to the natural language processing and machine learning areas, the first of all, is that people using social networks are kindly ignoring formal writing. For example, a typical Twitter user do not follow formal writing rules and introduces new lexical variations indiscriminately, the use of emoticons and the mix of languages is also the common lingo. These characteristics produce high dimensional representations, where the curse of dimension makes hard to learn from examples.

There exists a number of strategies to cope with the sentiment analysis on Twitter messages, some of them are based on the fact that the core problem is fixed: we are looking for evidence of some sentiment in the text. Under this scheme a number of dictionaries have been described by psychologists, other resources like SentiWordNet have been created adapting well known linguistic resources and machine learning. There is a lot of work around this approach; however, all these knowledge is language dependent and must exists a deep understanding of the language being analyzed. Our ap-

proach is mostly independent of this kind of external resources while focus on tackling the misspellings and other common errors in the text.

In this manuscript we detail our approach to sentiment analysis from a language agnostic perspective, e.g., no one in our team knows Italian language. We neither use external knowledge nor specialized parsers. Our aim is to create a solid baseline from a multilingual perspective, that can be used as a real baseline for challenges like SENTIPOLC'16 and as a basic initial approximation for sentiment analysis systems.

The rest of the paper is organized in the following sections. Section 2 describes our approach. Section 3 describes our experimental results, and finally Section 4 concludes.

2 Our participation

This participation is based on two approaches. First, B4MSA method, a simple approach which starts by applying text-transformations to the tweets, then transformed tweets are represented in a vector space model, and finally, a Support Vector Machine (with linear kernel) is used as the classifier. Second, B4MSA + EvoDAG, a combination of this simple approach with a Genetic programming scheme.

2.1 Text modeling with B4MSA

B4MSA is a system for multilingual polarity classification that can serve as a baseline as well as a framework to build sophisticated sentiment analysis systems due to its simplicity. The source code of B4MSA can be downloaded freely[1].

We used our previous work, B4MSA, to tackle the SENTIPOLC challenge. Our approach learns based on training examples, avoiding any digested knowledge as dictionaries or ontologies. This scheme allows us to address the problem without caring about the particular language being tackled.

The dataset is converted to a vector space using a standard procedure: the text is normalized, tokenized and weighted. The weighting process is fixed to be performed by TFIDF (Baeza-Yates and Ribeiro-Neto, 2011). After that process, a linear SVM (Support Vector Machines) is trained using 10-fold cross-validation (Burges, 1998). At the end, this classifier is applied to the test set to obtain the final prediction.

[1] https://github.com/INGEOTEC/b4msa

At a glance, our goal is to find the best performing normalization and tokenization pipelines. We state the modeling as a combinatorial optimization problem; then, given a performance measure, we try to find the best performing configuration among a large parameter space.

The list of transformations and tokenizers are listed below. All the text transformations considered are either simple to implement, or there is an open-source library (e.g. (Bird et al., 2009; Řehůřek and Sojka, 2010)) that implement it.

2.2 Set of Features

In order to find the best performing configuration, we used two sort of features that we consider them as parameters: cross-language and language-dependent features.

Cross-language Features could be applied in most similar languages and similar surface features. Removing or keeping *punctuation* (question marks, periods, etc.) and *diacritics* from the original source; applying or not applying the processes of *case sensitivity* (text into lowercase) and *symbol reduction* (repeated symbols into one occurrence of the symbol). *Word-based n-grams (n-words) Feature* are word sequences of words according to the window size defined. To compute the N-words, the text is tokenized and combined the tokens. For example, 1-words (unigrams) are each word alone, and its 2-words (bigrams) set are the sequences of two words, and so on (Jurafsky and Martin, 2009). *Character-based q-grams (q-grams)* are sequences of characters. For example, 1-grams are the symbols alone, 3-grams are sequences of three symbols, generally, given text of size m characters, we obtain a set with at most $m - q + 1$ elements (Navarro and Raffinot, 2002). Finally, *Emoticon (emo) feature* consists in keeping, removing, or grouping the emotions that appear in the text; popular emoticons were hand classified (positive, negative or neutral), included text emoticons and the set of unicode emoticons (Unicode, 2016).

Language Dependent Features. We considered three language dependent features: stopwords, stemming, and negation. These processes are applied or not applied to the text. *Stopwords* and stemming processes use data and the Snowball Stemmer for Italian, respectively, from NLTK Python package (Bird et al., 2009). *Negation* feature markers could change the polarity of the mes-

sage. We used a set of language dependent rules for common negation structures to attached the negation clue to the nearest word, similar to the approach used in (Sidorov et al., 2013).

2.3 Model Selection

The model selection, sometimes called hyper-parameter optimization, is the key of our approach. The default search space of B4MSA contains more than 331 thousand configurations when limited to multilingual and language independent parameters; while the search space reaches close to 4 million configurations when we add our three language-dependent parameters. Depending on the size of the training set, each configuration needs several minutes on a commodity server to be evaluated; thus, an exhaustive exploration of the parameter space can be quite expensive that makes the approach useless.

To reduce the selection time, we perform a stochastic search with two algorithms, *random search* and *hill climbing*. Firstly, we apply random search (Bergstra and Bengio, 2012) that consists on randomly sampling the parameter space and select the best configuration among the sample. The second algorithm consists on a *hill climbing* (Burke et al., 2005; Battiti et al., 2008) implemented with memory to avoid testing a configuration twice. The main idea behind hill climbing is to take a pivoting configuration (in our case we start using the best one found by random search), explore the configuration's neighborhood, and greedily moving to the best neighbor. The process is repeated until no improvement is possible. The configuration neighborhood is defined as the set of configurations such that these differ in just one parameter's value.

Finally, the performance of the final configuration is obtained applying the above procedure and cross-validation over the training data.

2.4 B4MSA + EvoDAG

In the polarity task besides submitting B4MSA which is a constrained approach, we decided to generate an unconstrained submission by performing the following approach. The idea is to provide an additional dataset that it is automatically label with positive and negative polarity using the Distant Supervision approach (Snow et al., 2005; Morgan et al., 2004).

We start collecting tweets (using Twitter stream) written in Italian. In total, we collect more than $10,000,000$ tweets. From these tweets, we kept only those that were consistent with the emoticon's polarity used, e.g., the tweet only contains consistently emoticons with positive polarity. Then, the polarity of the whole tweet was set to the polarity of the emoticons, and we only used positive and negative polarities. Furthermore, we decided to balance the set, and then we remove a lot of positive tweets. At the end, this external dataset contains $4,550,000$ tweets, half of them are positive and the another half are negative.

Once this external dataset was created, we decided to split it in batches of $50,000$ tweets half of them positive and the other half negative. This decision was taken in order to optimize the time needed to train a SVM and also around this number the Macro F1 metric is closed to its maximum value. That is, this number of tweets gives a good trade-off between time needed and classifier performance. In total there are 91 batches.

For each batch, we train a SVM at the end of this process we have 91 predictions (it is use the decision function). Besides these 91 predictions, it is also predicted (using as well the decision function) each tweet with B4MSA. That is, at the end of this process we have 94 values for each tweet. That is, we have a matrix with $7,410$ rows and 94 columns for the training set and of $3,000$ rows and 94 columns for the test set. Moreover, for matrix of the training set, we also know the class for each row. It is important to note that all the values of these matrix are predicted, for example, in B4MSA case, we used a 10-fold cross-validation in the training set in order to have predicted values.

Clearly, at this point, the problem is how to make a final prediction; however, we had built a classification problem using the decision functions and the classes provided by the competition. Thus, it is straight forward to tackle this classification problem using EvoDAG (Evolving Directed Acyclic Graph)[2] (Graff et al., 2017) which is a Genetic Programming classifier that uses semantic crossover operators based on orthogonal projections in the phenotype space. In a nutshell, EvoDAG was used to ensemble the outputs of the 91 SVM trained with the dataset automatically labeled and B4MSA's decision functions.

[2] https://github.com/mgraffg/EvoDAG

3 Results and Discussion

This Section presents the results of the INGEOTEC team. In this participation we did two runs, a constrained and an unconstrained run with B4MSA system, and only a constrained run with B4MSA + EvoDAG. The constrained run was conducted only with the dataset provided by SENTIPOLC'16 competition. For more technical details from the database and the competition in general see (Barbieri et al., 2016).

The unconstrained run was developed with an additional dataset of $4,550,000$ of tweets labeled with Distant Supervision approach. The Distant Supervision is an extension of the paradigm used in (Snow et al., 2005) and nearest to the use of weakly labeled data in (Morgan et al., 2004). In this case, we consider the emoticons as key for automatic labeling. Hence, a tweet with a high level of positive emoticons is labeled as positive class and a tweet with a clear presence of negative emoticons is labeled as negative class. This give us a bigger amount of samples for the dataset for training.

For the constrained run we participate in two task: subjectivity and polarity classification. In the unconstrained run we only participate in polarity classification task. Table 1 shows the results of subjectivity classification Task (B4MSA method), here, $\mathbf{Prec_0}$ is the $Precision_0$ value, $\mathbf{Rec_0}$ is the $Recall_0$ value, $\mathbf{FSc_0}$ is $F-Score_0$ value and $\mathbf{Prec_1}$, $\mathbf{Rec_1}$ and $\mathbf{FSc_1}$ the same for $F-Score_1$ values and $\mathbf{FSc_{avg}}$ is the average value from all F-Scores. The explanation of evaluation measures can be seen in (Barbieri et al., 2016).

Table 2, shows the results on the polarity classification task. In this task our B4MSA method achieves an average F-Score of 0.6054 and our combination of B4MSA + EvoDAG reaches an 0.6075 of average F-Score. These results place us on position 18 (unconstrained run) and 19 (constrained run) of a total of 26 entries.

It is important to mention that the difference between our two approaches is very small; however, B4MSA + EvoDAG is computationally more expensive, so we expected to have a considerable improvement in performance. It is evident that these results should be investigated further, and, our first impression are that our Distant supervision approach should be finely tune, that is, it is needed to verify the polarity of the emoticons and the complexity of the tweets.

Finally, Table 3 presents the measures employed by our internal measurement, that is Macro F1 and Micro F1 (for more details see (Sebastiani, 2002)). These values are from polarity unconstrained run (B4MSA + EvoDAG), polarity constrained run (B4MSA), subjectivity constrained run (B4MSA) and irony classification (B4MSA). We do not participate in irony classification task but we want to show the obtained result from our B4MSA approach on this task.

4 Conclusions

In this work we describe the INGEOTEC team participation in SENTIPOLC'16 contest. Two approaches were used, first, B4MSA method which combine several text transformations to the tweets. Secondly, B4MSA + EvoDAG, which combine the B4MSA method with a genetic programming approach. In subjectivity classification task, the obtained results place us in seventh of a total of 21 places. In polarity classification task, our results place us 18 and 19 places of a total of 26. Since our approach is simple and easy to implement, we take these results important considering that we do not use affective lexicons or another complex linguistic resource. Moreover, our B4MSA approach was tested internally in irony classification task with a result of 0.4687 of macro f1, and 0.8825 of micro f1.

References

Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 2011. *Modern Information Retrieval*. Addison-Wesley, 2nd edition.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In Pierpaolo Basile, Franco Cutugno, Malvina Nissim, Viviana Patti, and Rachele Sprugnoli, editors, *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. aAcademia University Press.

Roberto Battiti, Mauro Brunato, and Franco Mascia. 2008. *Reactive search and intelligent optimization*, volume 45. Springer Science & Business Media.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.

$\mathbf{Prec_0}$	$\mathbf{Rec_0}$	$\mathbf{FSc_0}$	$\mathbf{Prec_1}$	$\mathbf{Rec_1}$	$\mathbf{FSc_1}$	$\mathbf{FSc_{avg}}$
0.56	0.80	0.66	0.86	0.67	0.75	0.70

Table 1: Results on Subjectivity Classification

$\mathbf{FScore_{pos}}$	$\mathbf{FScore_{neg}}$	**Combined FScore**
Constrained run (B4MSA)		
0.6414	0.5694	0.6054
Unconstrained run (B4MSA + EvoDAG)		
0.5944	0.6205	0.6075

Table 2: Results on Polarity Classification

Run	Macro F1	Micro F1
Polarity Unconstrained	0.5078	0.5395
Polarity Constrained	0.5075	0.5760
Subjectivity Constrained	0.7137	0.721
Irony Constrained	0.4687	0.8825

Table 3: Micro F1 and Macro F1 results from our approaches

Christopher J.C. Burges. 1998. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167.

Edmund K Burke, Graham Kendall, et al. 2005. *Search methodologies*. Springer.

Mario Graff, Eric S. Tellez, Hugo Jair Escalante, and Sabino Miranda-Jimnez. 2017. Semantic Genetic Programming for Sentiment Analysis. In Oliver Schtze, Leonardo Trujillo, Pierrick Legrand, and Yazmin Maldonado, editors, *NEO 2015*, number 663 in Studies in Computational Intelligence, pages 43–65. Springer International Publishing. DOI: 10.1007/978-3-319-44003-3_2.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Alexander A. Morgan, Lynette Hirschman, Marc Colosimo, Alexander S. Yeh, and Jeff B. Colombe. 2004. Gene name identification and normalization using a model organism database. *Journal of Biomedical Informatics*, 37(6):396 – 410. Named Entity Recognition in Biomedicine.

G. Navarro and M. Raffinot. 2002. *Flexible Pattern Matching in Strings – Practical on-line search algorithms for texts and biological sequences*. Cambridge University Press. ISBN 0-521-81307-7. 280 pages.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May. ELRA. `http://is.muni.cz/publication/884893/en`.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, March.

Grigori Sidorov, Sabino Miranda-Jiménez, Francisco Viveros-Jiménez, Alexander Gelbukh, Noé Castro-Sánchez, Francisco Velásquez, Ismael Díaz-Rangel, Sergio Suárez-Guerra, Alejandro Treviño, and Juan Gordon. 2013. Empirical study of machine learning based approach for opinion mining in tweets. In *Proceedings of the 11th Mexican International Conference on Advances in Artificial Intelligence - Volume Part I*, MICAI'12, pages 1–14, Berlin, Heidelberg. Springer-Verlag.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304. MIT Press.

Unicode. 2016. Unicode emoji chart. `http://unicode.org/emoji/charts/full-emoji-list.html`. Accessed 20-May-2016.

Exploiting Emotive Features for the Sentiment Polarity Classification of tweets

Lucia C. Passaro, Alessandro Bondielli and **Alessandro Lenci**
CoLing Lab, Dipartimento di Filologia, Letteratura e Linguistica
University of Pisa (Italy)
lucia.passaro@for.unipi.it
alessandro.bondielli@gmail.com
alessandro.lenci@unipi.it

Abstract

English. This paper describes the CoLing Lab system for the participation in the constrained run of the EVALITA 2016 SENTIment POLarity Classification Task (Barbieri et al., 2016). The system extends the approach in (Passaro et al., 2014) with emotive features extracted from ItEM (Passaro et al., 2015; Passaro and Lenci, 2016) and FB-NEWS15 (Passaro et al., 2016).

Italiano. *Questo articolo descrive il sistema sviluppato all'interno del CoLing Lab per la partecipazione al task di EVALITA 2016 SENTIment POLarity Classification Task (Barbieri et al., 2016). Il sistema estende l'approccio descritto in (Passaro et al., 2014) con una serie di features emotive estratte da ItEM (Passaro et al., 2015; Passaro and Lenci, 2016) and FB-NEWS15 (Passaro et al., 2016).*

1 Introduction

Social media and microblogging services are extensively used for rather different purposes, from news reading to news spreading, from entertainment to marketing. As a consequence, the study of how sentiments and emotions are expressed in such platforms, and the development of methods to automatically identify them, has emerged as a great area of interest in the Natural Language Processing Community. Twitter presents many linguistic and communicative peculiarities. A tweet, in fact, is a short informal text (140 characters), in which the frequency of creative punctuation, emoticons, slang, specific terminology, abbreviations, links and hashtags is higher than in other domains and platforms. Twitter users post messages from many different media, including their smartphones, and they "tweet" about a great variety of topics, unlike what can be observed in other sites, which appear to be tailored to a specific group of topics (Go et al., 2009).

The paper is organized as follows: Section 2 describes the architecture of the system, as well as the pre-processing and the features designed in (Passaro et al., 2014). Section 3 shows the additional features extracted from emotive VSM and from LDA. Section 4 shows the classification paradigm, and the last sections are left for results and conclusions.

2 Description of the system

The system extends the approach in (Passaro et al., 2014) with emotive features extracted from ItEM (Passaro et al., 2015; Passaro and Lenci, 2016) and FB-NEWS15 (Passaro et al., 2016). The main goal of the work is to evaluate the contribution of a distributional affective resource to estimate the valence of words. The CoLing Lab system for polarity classification includes the following basic steps: (i) a preprocessing phase, to separate linguistic and nonlinguistic elements in the target tweets; (ii) a feature extraction phase, in which the relevant characteristics of the tweets are identified; (iii) a classification phase, based on a Support Vector Machine (SVM) classifier with a linear kernel.

2.1 Preprocessing

The aim of the preprocessing phase is the identification of the linguistic and nonlinguistic elements in the tweets and their annotation.

While the preprocessing of nonlinguistic elements such as links and emoticons is limited to their identification and classification (cf. section 2.2.4), the treatment of the linguistic material required the development of a dedicated rule-based procedure, whose output is a normalized text that is subsequently feed to a pipeline of general-purpose linguistic annotation tools. The following rules have been applied in the linguistic preprocessing phase:

- Emphasis: tokens presenting repeated characters like *bastaaaa* "stooooop" are replaced by their most probable standardized forms (i.e. *basta* "stop");
- Links and emoticons: they are identified and removed;
- Punctuation: linguistically irrelevant punctuation marks are removed;
- Usernames: the users cited in a tweet are identified and normalized by removing the @ symbol and capitalizing the entity name;
- Hashtags: they are identified and normalized by simply removing the # symbol;

The output of this phase are linguistically-standardized tweets, that are subsequently POS tagged with the Part-Of-Speech tagger described in (Dell'Orletta, 2009) and dependency-parsed with the DeSR parser (Attardi et al., 2009).

2.2 Feature extraction

The inventory of features can be organized into six classes. The five classes of features described in this section have been designed in 2014, the sixth class, described in the next section is referred to the emotive and LDA features.

2.2.1 Lexical Features

Lexical features represent the occurrence of bad words or of words that are either highly emotional or highly polarized. Relevant lemmas were identified from two in-house built lexicons (cf. below), and from Sentix (Basile and Nissim, 2013), a lexicon of sentiment-annotated Italian words. Lexical features include:

ItEM seeds: Lexicon of 347 highly emotional Italian words built by exploiting an online feature elicitation paradigm (Passaro et al., 2015). The features are, for each emotion, the total count of strongly emotional tokens in each tweet.

Bad words lexicon: By exploiting an in house built lexicon of common Italian bad words, we reported, for each tweet, the frequency of bad words belonging to a selected list, as well as the total amount of these lemmas.

Sentix: Sentix (Sentiment Italian Lexicon: (Basile and Nissim, 2013)) is a lexicon for Sentiment Analysis in which 59,742 lemmas are annotated for their polarity and intensity, among other information. Polarity scores range from -1 (totally negative) to 1 (totally positive), while Intensity scores range from 0 (totally neutral) to 1 (totally polarized). Both these scores appear informative for the classification, so that we derived, for each lemma, a Combined score C_{score} calculated as follows:

$$C_{score} = Intensity * Polarity \quad (1)$$

Depending on their C_{score}, the selected lemmas have been organized into several groups:

- strongly positives: $1 \leq C_{score} < 0.25$
- weakly positives:$0.25 \leq C_{score} < 0.125$
- neutrals: $0.125 \leq C_{score} \leq -0.125$
- weakly negatives: $-0.125 < C_{score} \leq -0.25$
- highly negatives: $-0.25 < C_{score} \leq -1$

Since Sentix relies on WordNet sense distinctions, it is not uncommon for a lemma to be associated with more than one ⟨Intensity,Polarity⟩ pair, and consequently to more than one C_{score}.

In order to handle this phenomenon, the lemmas have been splitted into three different ambiguity classes: Lemmas with only one entry or whose entries are all associated with the same C_{score} value, are marked as "Unambiguous" and associated with their C_{score}.

Ambiguous cases were treated by inspecting, for each lemma, the distribution of the associated C_{scores}: Lemmas which had a Majority Vote (MV) were marked as "Inferable" and associated with the $C_{score}$of the MV. If there was no MV, lemmas were marked as "Ambiguous" and associated with the mean of the C_{scores}. To isolate a reliable set of polarized words, we focused only on the Unambiguous or Inferable lemmas and selected only the

250 topmost frequent according to the PAIS corpus (Lyding et al., 2014), a large collection of Italian web texts.

Other Sentix-based features in the ColingLab model are: the number of tokens for each C_{score} group, the C_{score} of the first token in the tweet, the C_{score} of the last token in the tweet and the count of lemmas that are represented in Sentix.

2.2.2 Negation

Negation features have been developed to encode the presence of a negation and the morphosyntactic characteristics of its scope.

The inventory of negative lemmas (e.g. "non") and patterns (e.g. "non ... mai") have been extracted from (Renzi et al., 2001). The occurrences of these lemmas and structures have been counted an inserted as features to feed the classifier.

In order to characterize the scope of each negation, we used the dependency parsed tweets produced by DeSR (Attardi et al., 2009). The scope of a negative element is assumed to be its syntactic head or the predicative complement of its head, in the case the latter is a copula. Although it is clearly a simplifying assumption, the preliminary experiments show that this could be a rather cost-effective strategy in the analysis of linguistically simple texts like tweets.

This information has been included in the model by counting the number of negation patterns encountered in each tweet, where a negation pattern is composed by the PoS of the negated element plus the number of negative tokens depending from it and, in case it is covered by Sentix, either its Polarity, its Intensity and its C_{scores} value.

2.2.3 Morphological features

The linguistic annotation produced in the preprocessing phase has been exploited also in the population of the following morphological statistics: (i) number of sentences in the tweet; (ii) number of linguistic tokens; (iii) proportion of content words (nouns, adjectives, verbs and adverbs); (iv) number of tokens for Part-of-Speech.

2.2.4 Shallow features

This group of features has been developed to describe distinctive characteristics of web communication. The group includes:

Emoticons: We used the lexicon LexEmo to mark the most common emoticons, such as `:-(` and `:-)`, marked with their polarity score: 1 (positive), -1 (negative), 0 (neutral).

LexEmo is used both to identify emoticons and to annotate their polarity.
Emoticon-related features are the total amount of emoticons in the tweet, the polarity of each emoticon in sequential order and the polarity of each emoticon in reversed order. For instance, in the tweet *`:-(` (quando ci vediamo? mi manchi anche tu! `:*:*`* "`:-(` when are we going to meet up? I miss you, too `:*:*`" there are three emoticons, the first of which (`:-(`) is negative while the others are positive (`:*; :*`).
Accordingly, the classifier has been fed with the information that the polarity of the first emoticon is -1, that of the second emoticon is 1 and the same goes for the third emoticon. At the same way, another group of feature specifies that the polarity of the last emoticon is 1, as it goes for that of the last but one emoticon, while the last but two has a polarity score of -1.

Links: These features contain a shallow classification of links performed using simple regular expressions applied to URLs, to classify them as following: `video`, `images`, `social` and `other`. We also use as feature the absolute number of links for each tweet.

Emphasis: The features report on the number of emphasized tokens presenting repeated characters like *bastaaaa*, the average number of repeated characters in the tweet, and the cumulative number of repeated characters in the tweet.

Creative Punctuation: Sequences of contiguous punctuation characters, like `!!!`, `!?!?!?!!?!????!` or `.......`, are identified and classified as a sequence of dots, exclamations marks, question marks or mixed. For each tweet, the features correspond to the number of sequences belonging to each group and their average length in characters.

Quotes: The number of quotations in the tweet.

2.2.5 Twitter features

This group of features describes some Twitter-specific characteristics of the target tweets.

Topic: This information marks if a tweet has been retrieved via a specific political hashtag or keywords. It is provided by organizers as an attribute of the tweet;

Usernames: The number of @username in the tweet;

Hashtags: Hashtags play the role of organizing the tweets around a single topic, so that they are useful to be considered in determing their polarity (i.e. a tweet containing hashtags like and *#amore* "#love" and *#felice* "#happy" is expected to be positive and a tweet containing hashtags like *#ansia* "#anxiety" and *#stressato* "#stressedout" is expected to be negative. This group of features registers the presence of an hashtag belonging to the list of the hashtags with a frequency higher than 1 in the training corpus.

3 Introducing emotive and LDA features

In order to add emotive features to the CoLing Lab model, we created an emotive lexicon from the corpus FB-NEWS15 (Passaro et al., 2016) following the strategy illustrated in (Passaro et al., 2015; Passaro and Lenci, 2016). The starting point is a set of seeds strongly associated to one or more emotions of a given taxonomy, that are used to build centroid distributional vectors representing the various emotions.

In order to build the distributional profiles of the words, we extracted the list T of the 30,000 most frequent nouns, verbs and adjectives from FB-NEWS15. The lemmas in T were subsequently used as target and contexts in a square matrix of co-occurrences extracted within a five word window (± 2 words, centered on the target lemma). In addition, we extended the matrix to the nouns, adjectives and verbs in the corpus of tweets (i. e. lemmas not belonging to T).

For each $\langle$emotion, PoS$\rangle$ pair we built a centroid vector from the vectors of the seeds belonging to that emotion and PoS, obtaining in total 24 centroids[1]. Starting from these spaces, several groups of features have been extracted. The simplest ones include general statistics such as the number of emotive words and the emotive score of a tweet. More sophisticated features are aimed at inferring the degree of distinctivity of a word as well as its polarity from their own emotive profile.

Number of emotive words: Words belonging to the emotive Facebook spaces;

Emotive/words ratio: The ratio between the number of emotive words and the total number of words in the tweet;

Strongly emotive words: Number of words having a high (greater than 0.4) emotive score for at least one emotion;

Tweet emotive score: Score calculated as the ratio between the number of strongly polarized words and the number of the content words in the tweet (Eq. 2). The feature assumes values in the interval [0, 1]. In absence of strongly emotive words, the default value is 0.

$$E(Tweet) = \frac{Count(Strongly\ emotive\ words)}{Count(Content\ words)} \quad (2)$$

Maximum values: The maximum emotive value for each emotion (8 features);

Quartiles: The features take into account the distribution of the emotive words in the tweet. For each emotion, the list of the emotive words has been ordered according to the emotive scores and divided into quartiles (e.g. the fourth quartile contains the most emotive words and the first quartile the less emotive ones.). Each feature registers the count of the words belonging to the pair $\langle emotion, quartile \rangle$ (32 features in total);

ItEM seeds: Boolean features registering the presence of words belonging to the words used as seeds to build the vector space models. In particular, the features include the top 4 frequent words for each emotion (32 boolean features in total);

Distintive words: 32 features corresponding to the top 4 distinctive words for each emotion. The degree of distinctivity of a word for a given emotion is calculated starting from the VSM normalized using Z-scores. In particular, the feature corresponds to the proportion

[1]Following the configuration in (Passaro et al., 2015; Passaro and Lenci, 2016), the co-occurrence matrix has been re-weighted using the Pointwise Mutual Information (Church and Hanks, 1990), and in particular the Positive PMI (PPMI), in which negative scores are changed to zero (Niwa and Nitta, 1994). We constructed different word spaces according to PoS because the context that best captures the meaning of a word differs depending on the word to be represented (Rothenhusler and Schtze, 2007).

of the emotion $\langle emotion_i \rangle$ against the sum of total emotion score $[e_1, ..., e_8]$;

Polarity (count): The number of positive and negative words. The polarity of a word is calculated by applying Eq. 3, in which positive emotions are assumed to be JOY and TRUST, and negative emotions are assumed to be DISGUST, FEAR, ANGER and SADNESS.

$$Polarity(w) = \frac{\text{JOY+TRUST}}{2} - \frac{\text{DISGUST+FEAR+ANGER+SADNESS}}{4} \quad (3)$$

Polarity (values): The polarity (calculated using Eq. 3) of the emotive words in the tweet. The maximum number of emotive words is assumed to be 20;

LDA features: This group of features includes 50 features referred to the topic distribution of the tweet. The LDA model has been built on the FB-NEWS15 corpus (Passaro et al., 2016) which is organized into 50 clusters of thematically related news created with LDA (Blei et al., 2003) (Mallet implementation (McCallum, 2002)). Each feature refers to the association between the text of the tweet and a topic extracted from FB-NEWS15.

4 Classification

We used the same paradigm used in (Passaro et al., 2014). In particular, we chose to base the CoLing Lab system for polarity classification on the SVM classifier with a linear kernel implementation available in Weka (Witten and Frank, 2011), trained with the Sequential Minimal Optimization (SMO) algorithm introduced by Platt (Platt, 1999).

The classification task proposed by the organizers could be approached either by building two separate binary classifiers relying of two different models (one judging the positiveness of the tweet, the other judging its negativeness), or by developing a single multiclass classifier where the possible outcomes are Positive Polarity (Task POS:1, Task NEG:0), Negative Polarity (Task POS:0, Task NEG:1), Mixed Polarity (Task POS:1, Task NEG:1) and No Polarity (Task POS:0, Task NEG:0). In Evalita 2014 (Passaro et al., 2014) we tried both approaches in our development phase, and found no significant difference, so that we opted for the more economical setting, i.e. the multiclass one.

5 Results

Although this model is not optimal according to the global ranking, if we focus on the recognition of the negative tweets (i.e. the NEG task), it ranks fifth (F1-score), and first if we consider the class 1 of the NEG task (i.e. NEG, F.sc. 1). Such trend is reversed if we consider the POS task, which is the worst performing class of this system.

Task	Class	Precision	Recall	F-score
POS	0	0,8548	0,7682	0,8092
POS	1	0,264	0,3892	0,3146
POS task		0,5594	0,5787	0,5619
NEG	0	0,7688	0,6488	0,7037
NEG	1	0,5509	0,6883	0,612
NEG task		0,65985	0,66855	0,6579
GLOBAL		0,609625	0,623625	0,6099

Table 1: System results.

Due to the great difference in terms of performance between the results obtained by performing a 10 fold cross validation, we suspected that the system was overfitting the training data, so that we performed different feature ablation experiments, in which we included only the lexical information derived from ItEM and FB-NEWS (i.e. we removed the features relying to Sentix, Negation and Hashtags (cf. table 2). The results demonstrate on one hand that significant improvements can be obtained by using lexical information, especially to recognize negative texts. On the other hand the results highlight the overfitting of the submitted model, probably due to the overlapping between Sentix and the emotive features.

Task	Class	Precision	Recall	F-score
POS	0	0,8518	0,8999	0,8752
POS	1	0,3629	0,267	0,3077
POS task		0,60735	0,58345	0,59145
NEG	0	0,8082	0,6065	0,693
NEG	1	0,5506	0,7701	0,6421
NEG task		0,6794	0,6883	0,66755
GLOBAL		0,643375	0,635875	0,6295

Table 2: System results for a filtered model.

The advantage of using only the lexical features derived from ItEM are the following: i) the emotional values of the words can be easily updated; ii) the VSM can be extended to increase the lexical coverage of the resource; iii) the system is "lean" (it can do more with less).

6 Conclusions

The Coling Lab system presented in 2014 (Passaro et al., 2014) has been enriched with emotive features derived from a distributional, corpus-based resource built from the social media corpus FB-NEWS15 (Passaro et al., 2016). In addition, the system exploits LDA features extacted from the same corpus. Additional experiments demonstrated that removing most of the non-distributional lexical features derived from Sentix, the performance can be improved. As a consequence, with a relatively low number of features the system reaches satisfactory performance, with top-scores in recognizing negative tweets.

References

Giuseppe Attardi, Felice Dell'Orletta, Maria Simi, and Joseph Turian. 2009. Accurate dependency parsing with a stacked multilayer perceptron. In *Proceedings of EVALITA 2009 Evaluation of NLP and Speech Tools for Italian 2009*, Reggio Emilia (Italy). Springer.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In *Proceedings of EVALITA 2016 Evaluation of NLP and Speech Tools for Italian*, Napoli (Italy). Academia University Press.

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Kenneth W. Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16:22–29.

Felice Dell'Orletta. 2009. Ensemble system for part-of-speech tagging. In *Proceedings of EVALITA 2009 Evaluation of NLP and Speech Tools for Italian 2009*, Reggio Emilia (Italy). Springer.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing*, pages 1–6.

Verena Lyding, Egon Stemle, Claudia Borghetti, Marco Brunello, Sara Castagnoli, Felice DellOrletta, Henrik Dittmann, Alessandro Lenci, and Vito Pirrelli. 2014. The PAISÀ Corpus of Italian Web Texts. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, pages 36–43, Gothenburg (Sweden). Association for Computational Linguistics.

Andrew K. McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Yoshiki Niwa and Yoshihiko Nitta. 1994. Co-occurrence vectors from corpora vs. distance vectors from dictionaries. In *Proceedings of the 15th International Conference On Computational Linguistics*, pages 304–309, Kyoto (Japan).

Lucia C. Passaro and Alessandro Lenci. 2016. Evaluating context selection strategies to build emotive vector space models. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Portoro (Slovenia). European Language Resources Association (ELRA).

Lucia C. Passaro, Gianluca E. Lebani, Emmanuele Chersoni, and Alessandro Lenci. 2014. The coling lab system for sentiment polarity classification of tweets. In *Proceedings of the First Italian Conference on Computational Linguistics CLiC-it 2014 & and of the Fourth International Workshop EVALITA 2014*, pages 87–92, Pisa (Italy).

Lucia C. Passaro, Laura Pollacci, and Alessandro Lenci. 2015. Item: A vector space model to bootstrap an italian emotive lexicon. In *Proceedings of the second Italian Conference on Computational Linguistics CLiC-it 2015*, pages 215–220, Trento (Italy).

Lucia C. Passaro, Alessandro Bondielli, and Alessandro Lenci. 2016. Fb-news15: A topic-annotated facebook corpus for emotion detection and sentiment analysis. In *Proceedings of the Third Italian Conference on Computational Linguistics CLiC-it 2016.*, Napoli (Italy). To appear.

John C. Platt, 1999. *Advances in Kernel Methods*, chapter Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pages 185–208. MIT Press, Cambridge, MA, USA.

Lorenzo Renzi, Giampaolo Salvi, and Anna Cardinaletti. 2001. *Grande grammatica italiana di consultazione*. Number v. 1. Il Mulino.

Klaus Rothenhusler and Hinrich Schtze. 2007. Part of speech filtered word spaces. In *Sixth International and Interdisciplinary Conference on Modeling and Using Context*.

Ian H. Witten and Eibe Frank. 2011. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

Samskara
Minimal structural features for detecting subjectivity and polarity in Italian tweets

Irene Russo, Monica Monachini
Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC CNR)
Lari Lab
firstname.secondname@ilc.cnr.it

Abstract

English. Sentiment analysis classification tasks strongly depend on the properties of the medium that is used to communicate opinionated content. There are some limitations in Twitter that force the user to exploit structural properties of this social network with features that have pragmatic and communicative functions. Samskara is a system that uses minimal structural features to classify Italian tweets as instantiations of a textual genre, obtaining good results for subjectivity classification, while polarity classification needs substantial improvements.

Italiano. *I compiti di classificazione a livello di sentiment analysis dipendono fortemente dalle proprietà del mezzo usato per comunicare contenuti d'opinione. Vi sono limiti oggettivi in Twitter che forzano l'utente a sfruttare le proprietà strutturali del mezzo assegnando ad alcuni elementi funzioni pragmatiche e comunicative. Samskara è un sistema che si propone di classificare i tweets italiani come se appartenessero a un genere testuale, interprentandoli come elementi caratterizzati da strutture minimali e ottenendo buoni risultati nella classificazione della soggettività mentre la classificazione della polarità ha bisogno di sostanziali miglioramenti.*

1 Introduction

After 15 years of NLP works on the topic Sentiment Analysis is still a relevant task, mainly because we assist every day to an exponential growth of opinionated content on the web that require computational systems to be managed. Detected, extracted and classified, opinionated content can also be labeled as positive or negative, but additional categories (ambiguous, neutral etc.) are possible. Resources and methodologies created for the detection and classification of subjectivity and polarity in reviews are not applicable with good results on different data, such as tweets or comments about news from online fora.

There are several reasons behind this: first and foremost, opinions can be expressed more or less explicitly depending on the context; lexical cues from lexical resources such as SentiWordNet (Baccianella et al., 2010) or General Inquirer (Stone, 1966) could be useless when people write their point of views in complex and subtle ways. Secondly, different media and platforms impose different constraints on the structure of the content expressed.

Twitter's limits in terms of characters force the use of abbreviations and the omission of syntactic elements. But users try to exploit creatively these limitations, for example adding pragmatic functions with emoticons.

Features and functionalities anchoring the text to extra-linguistic dimensions (such as mentions and pictures in tweets or *like/agree* from other users in online debates) should be considered in Sentiment Analysis classification tasks because of to their communicative functions.

In this paper we present Samskara, a Lari lab system for the classification of Italian tweets that took part in two tasks at Sentipolc2016 (Task 1,subjectivity and Task 2, polarity classification). The system is described in par. 2, with results presented in 2.2 where we discuss the limitations of the system.

2 System description

Samskara is a classification system based on a minimal set of features that wants to address the issue of subjectivity and polarity classifications of Italian tweets. Tweets are considered as instantiations of a textual genre, namely they have specific structural properties with communicative and pragmatic functions. In our approach, focusing on the structural properties means:

- abstracting the task from lexical values of single words that could be a deceptive cue because of lexical sparseness, ambiguity of words, use of jargon and ironic exploitations of words;
- taking into account features used in authorship attribution to represent abstract patterns characterizing different styles, e.g. PoS tag n-gram frequencies(Stamatos, 2009)[1];
- choosing a tagset for PoS that includes tags peculiar of tweets as a textual genre, i.e. interjection and emoticon.

More generally, we want to capture high-level linguistic and extra-linguistic properties of tweets, also considering basic sequential structures in forms of sequences of bigrams.

2.1 Data analysis, data preprocessing and feature selection

Before starting with the selections of features, data analysis of the training set helped in the investigation of several hypotheses.
Polarised lexical items have been widely used in sentiment analysis classification (Liu and Zhang, 2012) but resources in this field list values at sense level (such as SentiWordNet) or conflate the senses in a single entry (such as General Inquirer and LIWC). Without an efficient word sense disambiguation module, using SentiWordNet is difficult. One strategy is to sum all the values and to select a threshold for words that are tagged as polarised in text. That means to overstimate positive/negative content, without finding a clear boundary between, for example, positive and negative tweets.
Considering the Italian version of LIWC2015 (Pennebaker et al., 2015) we see that frequencies are unable to distinguish between positive and negative tweets in the Sentipolc2016 training data (see Table 1). To avoid this, we defined for internal use a subset of SentiWordNet 3.0 (Baccianella et al., 2010) that we call SWN Core selecting:

class	tokens	LIWC+	LIWC-
pos	92295	234 (0.26%)	225 (0.25%)
neg	114435	78 (0.07%)	683 (0.6%)

Table 1: Absolute and relative frequencies of Italian LIWC2015 lemmas in positive and negative tweets (Sentipolc2016 training set).

- all the words corresponding to senses that are polarised;
- from the set above, all the words corresponding to senses that display single-valued polarity (i.e. they are always positive or always negative);
- from the set above we delete all the words that have also a neutral sense;
- we sum polarity values for every lemma in order to have for example a single value for lemmas listed in SWN with two different positive values or three different negative values.

The English SWN Core is composed by 6640 exclusively positive lemmas and 7603 exclusively negative lemmas. Since in these lists items have a polarity value ranging from 0.125 to 3.25, with the idea of selecting lemmas that are strongly polarised we set 0.5 as threshold; as a consequence of this decision we have 1844 very positive and 3272 very negative lemmas. After deletion of multiword expressions these strongly opinionated words have been translated to Italian using Google Translate, manually checked and annotated with PoS and polarity.
We clean the lists, deleting lemmas that appear two times, lemmas that have been translated as multiword expressions and lemmas that do not have polarity in Italian. At the end we have 890 positive and 1224 negative Italian lemmas. Considering their frequencies in the training set (see Table 2) we find out that only negative items are distinctive. Because of the presence of ironic tweets positive lemmas tend to occur in tweets that

[1]For the moment we think that sequences of syntactic relations are not useful because of the poor performance of Italian syntactic parsers on tweets.

have been tagged as negative. The exploitation of positive words in ironic communication is a well-known phenomenon (Dews and Winner, 1995) - the positive literal meaning is subverted by the negative intended meaning - and neglecting this aspect of the Sentipolc2016 training set could imply lower classification performances. If we allow positive items from SWN Core in the system the classification of negative tweets is made difficult. As we mention above, structural properties

	SWN Core+	SWN Core-
obj	536 (0.76%)	264 (0.37%)
subj	2307 (1.4%)	1608 (1%)
pos	1055 (4.8%)	200 (0.9%)
neg	839 (2%)	1096 (2.6%)

Table 2: Absolute and relative frequencies of SWN Core lemmas in Sentipolc2016 training set.

of tweets can be treated as sequences of PoS. To reduce data sparseness and to include dedicated tags for Twitter we choose the tagset proposed by PoSTWITA, an Evalita2016 task (Bosco et al., 2016). It looks promising because it contains categories that:

- could be easily tagged as preprocessing step with regular expressions (for example MENTION and LINK);
- are suitable for noisy data, tagging uniformly items that can be written in several, non-predictable ways (*ahahahha, haha* as INTJ);
- contains tags that have communicative and pragmatic functions, such as emoticon and interjection (see Table 4).

We preprocessed all the tweets in the training set substituting elements that are easy to find, such as mention, hashtags, email, link, emoticon (all tags included in PoSTWITA).
After that, Sentipolc2016 training set has been tagged with TreeTagger (Schmid, 1997); TreeTagger tags have been converted to PostTWITA tagset (see Table 3) and additional tags from PosTWITA have been added, building dedicated lists for them that include items from PoSTWITA training set plus additional items selected by the authors (see Table 4).
Thanks to TreeTagger we have all the words lemmatized and so all the lemmas included in the negative counterpart of SWN Core can be substituted

TreeTagger	PoSTWITA
AUX	[A-Z a-z]+ AUX
DET	[A-Z a-z]+ DET
PRO	[A-Z a-z]+ PRON
NPR	[A-Z a-z]+ PROPN
PUN	PUNCT
SENT	PUNCT
VER[A-Z a-z]+cli	VERB_CLIT
VER	[A-Z a-z]+ VERB

Table 3: Comparison between TreeTagger and PoSTWITA tagsets.

by the tag VERYNEG. At this point, with the intention to have a minimal sequence of significant tags, we created 4 version of the training set according to 4 minimal structures, deleting all lemmas and leaving only PoS tags:

- minimal structure 1 (MSTRU1): EMO, MENTION, HASHTAG, URL, EMAIL;
- minimal structure 2 (MSTRU2): EMO, MENTION, HASHTAG, URL, EMAIL, PROPN, INTJ;
- minimal structure 3 (MSTRU3): EMO, MENTION, HASHTAG, URL, EMAIL, PROPN, INTJ, ADJ, ADV;
- minimal structure 4 (MSTRU4): EMOTICON, MENTION, HASHTAG, URL, EMAIL, PROPN, INTJ, VERYNEG.

We performed classification experiments with these features and we get better results with MSTRU4 (see par. 2.2).
For Samskara each tweet is represented as a sequence including its EMO, MENTION, HASHTAG, URL, EMAIL, PROPN (Proper Noun), INTJ and VERYNEG lemmas from SWN Core (see tweet in example 1 represented in example 2). This minimal, very compact way to represent a tweet is very convenient because partially avoids any noise introduced by PoS tagger (containing only VERYNEG and PROPN as elements that should be properly tagged with this tool).

(1) @FGoria Mario Monti Premier! #Italiaresiste.

(2) MENTION PROPN HASHTAG.

Additional features for the classification of subjective and positive or negative tweets are listed in

new tag	type	examples
PART	particle	*'s*
EMO	emoticon	*:DD, :-))))*, *u___u*
INTJ	interjection	*ah, boh, oddioo*
SYM	symbol	*%, &, <*
CONJ	coordinating conjunction	*ebbene, ma, oppure*
SCONJ	subordinating conjunction	*nonostante, mentre, come*

Table 4: Examples of lemmas tagged according to Twitter-specific PoSTWITA tags.

Table 5, with BOOL meaning boolean feature and NUM numeric feature (they correspond to absolute frequencies). The features have been selected thinking about their communicative function: *a1* for example is useful because there is a tendency to communicate opinionated content in discussions with other users while we choose *a2* because neutral tweets often advertise newspapers' articles in a non opinionated way including the link at the end of the tweet, but the URL is significant in other positions *a6, a6_1*. Together with emoticons, interjections are items that signal the presence of opinionated content. For the kind of asynchronous communication that characterize them, tweets can contain questions that don't expect an answer, that are rethorical *a8_1*, thus making the tweet opinionanted.

2.2 Results and Discussion

The system adopts the Weka[2] library that allows experiments with different classifiers. Due to better performance of Naive Bayes (default settings, 10- fold cross validation) with respect to Support Vector Machine we choose the first; best performances were obtained with MSTRU4 considering frequencies of unigrams and bigrams of PoS as features. We took part to Sentipolc2016 only with a constrained run, choosing slightly different set of features for subjectivity and polarity evaluation.
Adding the additional features in Table 5 we selected for Task 1 a subset of them after an ablation test. More specifically, the feature set 1 (FS1 in Table 7) is composed by features *a1, a2, a4, a4_1, a6, a6_1, a7, a7_1, a8_1, a9*. The system performance is reported in terms of F-score, according to the measure adopted by the task organizers (Barbieri et al., 2016). Results on the training data look promising for Task 1, less promising for Task 2 (see Table 8). We didn't succeed in optimising features for the polarity detection sub-task. The performance on the training set was not satisfying but nevertheless we decided to submit results for Task 2 on test set using all the features. In Table 9 the official results submitted for the competition are reported. Samskara was first among the constrained systems for subjectivity classification, while not surprisingly the performance in Task 2 was bad. Results in Task 2 can be explained by the absence in the system of structural features that are meaningful for the positive-negative distinctions or by the unsuitability of such a minimal approach for the task. It is possible that richer semantic features are necessary for the detection and the classification of polarity and polarised lexical items should be revised, for example, representing each lemma as a sentiment specific word embedding (SSWE) encoding sentiment information (Tang et al., 2014).
With Samskara we prove that classification of tweets should take into account structural properties of content on social media, especially properties that have communicative and pragmatic functions. The minimal features we selected for Samskara were successful for the classification of subjective Italian tweets. The system is based on a minimal set of features that are easy to retrieve and tag; the classification system is efficient and fast for Task 1 and as such it is promising for real-time processing of big data stream.

[2]http://www.cs.waikato.ac.nz/ml/weka/

References

Stefano Baccianella and Andrea Esuli and Fabrizio Sebastian. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Barbieri, Francesco and Basile, Valerio and Croce, Danilo and Nissim, Malvina and Novielli, Nicole and Patti, Viviana. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification

features	description	type
a1	the tweet starts with MENTION	BOOL
a2	the tweet ends with a LINK	BOOL
a3	the tweet has PoS of type PUNCT	BOOL
a3_1	number of PoS of type PUNCT in each tweet	NUM
a4	the tweet has PoS of type VERYNEG	BOOL
a4_1	number of PoS of type VERYNEG in each tweet	NUM
a5	the tweet has PoS of type INTJ	BOOL
a5_1	number of PoS of type INTJ in each tweet	NUM
a6	the tweet has PoS of type URL	BOOL
a6_1	number of PoS of type URL in each tweet	NUM
a7	the tweet has PoS of type EMOTICON	BOOL
a7_1	number of PoS of type EMOTICON in each tweet	NUM
a8_1	the tweet contains a question	BOOL
a8_2	the tweet contains a question at the end	BOOL
a9	the tweet contains two consecutive exclamation marks ('!!')	BOOL
a10	the tweets contains connectives such as *anzitutto, comunque, dapprima, del resto*	BOOL

Table 5: Additional features for subjectivy and polarity classification of tweets.

	MSTRU4 + FS1
obj F-score	0.532
subj F-score	0.811
Avg F-score	0.724

Table 6: Classification results for Task 1 obtained on Sentipolc2016 training set.

	MSTRU4 + AllF
pos F-score	0.424
neg F-score	0.539
both F-score	0.047
neu F-score	0.526
Avg F-score	0.48

Table 7: Classification results for Task 2 obtained on Sentipolc2016 training set.

	F-score	**Rank**
Task 1	0.7184	1
Task 2	0.5683	13

Table 8: Classification results for Task 1 and Task 2 on Sentipolc2016 test set.

Task. In *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*.

Bosco, Cristina and Tamburini, Fabio and Bolioli, Andrea and Mazzei, Alessandro. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian Task. In *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*.

Shelly Dews and Ellen Winner. 1995. Muting the meaning: A social function of irony. *Metaphor and Symbolic Activity*, 10(1):319.

Bing Liu and Lei Zhang. 2012. A Survey of Opinion Mining and Sentiment Analysis. In C. C. Aggarwal & C. Zhai (Eds.) *Mining Text Data*, pp. 415–463, US: Springer.

James W. Pennebaker, Ryan L. Boyd, Kayla Jordan, and Kate Blackburn. 2015. *The Development and Psychometric Properties of LIWC2015*.

Helmut Schmid. 1997. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *New Methods in Language Processing*, UCL Press, pp. 154-164.

Efstathios Stamatatos. 2009. A Survey of Modern Authorship Attribution Methods. *Journal of the American Society for Information Science and Technology*.

Stone, Philip J. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. The MIT Press.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu and Bing Qin. 2014. Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.