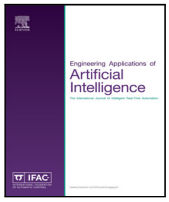
Engineering Applications of Artificial Intelligence 91 (2020) 103568

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



An encoder–decoder approach to mine conditions for engineering textual data☆

Fernando O. Gallego *, Rafael Corchuelo

Universidad de Sevilla, ETSI Informática, Avda. de la Reina Mercedes, s/n. Sevilla E-41012, Spain

ARTICLE INFO

Keywords:
Condition mining
Natural language processing
Neural networks

ABSTRACT

Data engineering seeks to support artificial intelligence processes that extract knowledge from raw data. Many such data are rendered in natural language from which entity-relation extractors extract facts and opinion miners extract opinions; the goal of condition mining is to mine the conditions that have an influence on them. In this article, a new condition mining method is proposed. It relies on a deep neural network and attempts to overcome the limitations of existing methods for condition mining that we reviewed. The materials used include readily-available software components for natural language processing and a large multi-lingual, multi-topic dataset. The common information retrieval performance measures were used to assess the results, namely: precision, which is the fraction of correct conditions to the mined ones, recall, which is the fraction of correct conditions that have been mined to the total number of correct conditions, and the F_1 score, which is the harmonic mean of precision and recall. The results of the experimental analysis prove that the new proposal can attain an F_1 score that is significantly greater than with existing methods. Furthermore, a comprehensive analysis of the dataset was performed, which revealed two key findings: the connectives follows a long-tail distribution and the conditions are quite dissimilar from a semantic point of view.

1. Introduction

Data engineering is an interdisciplinary field that uses scientific methods, hardware systems, and software agents to extract knowledge from data that is gathered from a variety of sources. Many such sources are unstructured because they render data in natural language without explicit computer-readable semantics. Such sources must be pre-processed using entity-relation extractors and opinion miners so as to extract their data in a structured format that is amenable for further processing. Entity-relation extractors focus on factual data and opinion miners focus on subjective data. That is, they provide a foundation that helps data engineering benefit from unstructured data sources.

Many entity-relation extractors (Etzioni et al., 2011; Velásquez et al., 2011; Yang and Soo, 2012; Vicient et al., 2013; Mitchell et al., 2015; Lima et al., 2019) and opinion miners (Perikos and Hatzilygeroudis, 2016; Jin et al., 2016a,b; Zhang et al., 2016; Pablos et al., 2018; Yoo et al., 2018; Ducange et al., 2019) have been analysed in this article. The conclusion is that they have a common problem: they do not take conditions into account. A condition is a clause that describes the circumstances or factors that must be met for something else to hold. For instance, a sentence like "may the new law be approved so that Acme Bank merges Trust Bank" asserts the fact that "Acme Bank merges Trust Bank" as long as condition "the new law is approved" holds; similarly, a sentence like "the lens is good enough for amateurs" conveys a positive opinion about "the lens" as long as condition "for amateurs" holds. Note that mining conditions is paramount to interpreting the previous fact or opinion correctly. The estimation is that roughly 10% of the sentences in the 4.7M sentence dataset that was compiled for this article have at least one condition. Simply put, unless those conditions are mined and returned with their corresponding facts and opinions, the quality of the succeeding data engineering processes will be cheapen.

Unfortunately, the frameworks (Bank and Schierle, 2012; Cunningham et al., 2013) and toolkits (Manning et al., 2014; Padró and Stanilovsky, 2012; Bird et al., 2009; The Apache Software Foundation, 2018; Honnibal and Montani, 2018) that support current entity-relation extractors or opinion miners do not provide any components to mine conditions. In the literature, there are two methods that require user-defined patterns on some common grammatical landmarks (Mausam et al., 2012; Chikersal et al., 2015); such handcrafted approaches may result in high precision, but fall short regarding recall because condition connectives have a long-tail distribution, which implies that there are many common conditions that do not fit common patterns. There is also a machine-learning method (Nakayama and Fujii, 2015), but it must

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.engappai.2020.103568.

* Corresponding author.
E-mail addresses: fogallego@us.es (F.O. Gallego), corchu@us.es (R. Corchuelo).

https://doi.org/10.1016/j.engappai.2020.103568
Received 10 March 2019; Received in revised form 7 January 2020; Accepted 15 February 2020
Available online 29 February 2020

be customised with several specific-purpose dictionaries, taxonomies, and heuristics, it mines conditions regarding opinions only, it is tightly bound to the Japanese language, and it was evaluated on a very small dataset.

In this article, a condition mining method is proposed. It jumps ahead the following scientific gap: it is possible to mine conditions from text that is written in natural language. The method relies on a deep neural approach (Han et al., 2017) that does not require any user-defined patterns, does not require any specific-purpose resources, can mine conditions in both factual and opinion sentences, and uses readily-available components only (a stemmer and a word embedder). An experimental analysis was performed on a large dataset with 4.7M sentences on 16 common topics in 4 common languages; the results prove that the new approach beats the others in terms of F_1 score. Summing up, it is a promising approach to improve data engineering when entity-relation extractors and opinion miners are involved.

The rest of the article is organised as follows: Section 2 reports on the related work; Section 3 introduces some preliminary concepts; Section 4 describes the new method to mine conditions; Section 5 reports on the experimental analysis; finally, Section 6 presents some conclusions.

2. Related work

This section first analyses current text mining systems, then focuses on the literature on condition mining, and finally, discusses on the findings.

2.1. Text mining systems

There are several entity-relation extraction systems in the literature. Etzioni et al. (2011) described a system to extract verb-based relations. It is based on two types of constraints, namely: syntactical, which are composed of part-of-speech regular expressions that reduce incoherences, and lexical, which build on a large dictionary of relations. Velásquez et al. (2011) described a system to extract key objects from web sites. The idea is to extract new relations between the structured components in a web site, which are represented by a simple core ontology. Relations are extracted by means of a set of patterns that are collected from several sessions with web users. Yang and Soo (2012) described a system to extract conceptual graphs from patent claims. The system works on a finite state machine that splits a patent claim sentence into a set of shorter sub-sentences that are then parsed. The POS and dependency tree of a patent claim are used to build the conceptual graph based on a pre-established domain ontology. Mitchell et al. (2015) described a complete architecture for entity-relation extraction. It applies several methods to extract entity-relations to the same piece of text and computes a level of confidence. Lima et al. (2019) described a logic-based relational learning approach that uses inductive logic programming to learn symbolic extraction rules. It uses a domain ontology that guides the background knowledge generation process and is used for storing the extracted relation instances.

There are also several systems in the literature on opinion mining. Perikos and Hatzilygeroudis (2016) presented an opinion analysis system for automatic recognition of emotions in text. It is based on an ensemble of three classifiers: a Naive Bayes learner, a Maximum Entropy learner, and a knowledge-based tool that performs a deep analysis of the sentences. The knowledge-based tool analyses the dependency structure of the sentence and implements a keyword-based approach in which the emotional state is derived from the emotional affinity of the emotional parts of the sentence. Jin et al. (2016a) presented a system to select pairs of opinionated representative yet comparative sentences with specific product features from reviews of competitive products. With the help of some opinion analysis techniques, opinionated sentences that refer to a specific feature are first identified from on-line reviews. Then, information representativeness, information comparativeness, and information diversity are investigated for the selection of a small number of representative comparative opinionated sentences. Accordingly, an optimisation problem is formulated, and three greedy algorithms are proposed to analyse this problem so as to find suboptimal solutions. Jin et al. (2016b) presented a system to identify product features and analyse opinions with the help of pro and con reviews. Conditional random fields were employed to detect aspects of product features and detailed reasons from on-line reviews jointly. Furthermore, a co-clustering algorithm was devised to group similar aspects and reasons. Zhang et al. (2016) presented an opinion mining extraction system to jointly discover the main opinion elements in a piece of text. It automatically builds kernels to combine closely related words into new terms from word level to phrase level based on dependency relations; the accuracy of opinion expressions and polarity is ensured based on fuzzy measurements, opinion degree intensifiers, and opinion patterns. Pablos et al. (2018) presented an almost unsupervised system that is based on topic modelling for aspect-based opinion analysis. It is combined with some other unsupervised methods and a minimal configuration. It performs aspect and category classification, aspect-term/opinion-word separation and opinion classification for any given domain and language.

Regarding text-mining frameworks, UIMA (Bank and Schierle, 2012) and GATE (Cunningham et al., 2013) range amongst the most popular ones. They both help design and implement pipelines in which each stage gets a message and the annotations produced by the previous stages, if any, as input and computes further annotations as output. The annotations range from the word stems or part-of-speech tags of the input words to their dependency tags and opinion scores, to mention a few. They both found their way into the text-mining field since they provide many off-the-shelf components, there are many third-party components, and they are open to integrate custom in-house components. They both support multiple languages and can read text from a variety of sources, including plain text, HTML documents, PDF documents, or databases. There are also many toolkits that provide components to implement specific natural-language-processing tasks, e.g., Stanford Core NLP (Manning et al., 2014), Freeling (Padró and Stanilovsky, 2012), and NLTK (Bird et al., 2009). Recently, toolkits like OpenNLP (The Apache Software Foundation, 2018) and spaCy (Honnibal and Montani, 2018) have gained popularity thanks to their focus on efficiency without sacrificing effectiveness.

Many of the previous scientific results have found their way into commercial text-mining services, e.g., Lithium, Sprout Social, Lexalytics, Brand Watch, Sysomos, or Opileak. They all can analyse text from sources like blogs, social networks, customer review sites, or news sites, just to mention a few; they can extract topics, identify their aspects, assess the opinion cast on them, or compute volumetric insights.

2.2. Literature on condition mining

The importance of mining conditions was first highlighted by Narayanan et al. (2009) in the field of opinion mining. Their method is a machine-learning model to compute whether a conditional sentence conveys a positive, a negative, or a neutral opinion. Unfortunately, they assumed that the sentences were previously labelled so as to make the conditions explicit and they did not provide their dataset. Recently, Skeppstedt et al. (2015) presented a complementary method that can automatically classify a sentence as speculative, contrast, or conditional, but neither was their goal to mine conditions. They worked on a modified version of the SFU Review Corpus[1] that included such categories; unfortunately, their version of the dataset was not published.

[1] The dataset is available at https://www.sfu.ca/~mtaboada/SFU_Review_Corpus.

The naivest methods to mine conditions are based on searching for user-defined patterns with syntactical anchors. Mausam et al. (2012) studied the problem in the field of entity-relation extraction; they realised that many usual conditions can be identified by locating adverbial clauses whose first word is one of the sixteen one-word condition connectives in English; unfortunately, they did not report on the effectiveness of their method to mine conditions, only on the overall effectiveness of their entity-relation extractor. Their system was updated recently with new features (Mausam, 2016), but their method to mine conditions was not. Chikersal et al. (2015) reported on a similar, but simpler method: search for sequences of words in between connectives "if", "unless", "until", and "in case" and the first occurrence of word "then" or a comma. Unfortunately, none of the previous proposals provide a publicly-available dataset.

The only existing machine-learning method was introduced by Nakayama and Fujii (2015), who worked in the field of opinion mining in Japanese. They devised a model that is based on features that are computed by means of a syntactical parser and a semantic analyser. The former identifies so-called bunsetus, which are Japanese syntactical units that consists of one independent word and one or more ancillary words, as well as their inter-dependencies; the latter identifies opinion expressions, which requires to provide some specific-purpose dictionaries, taxonomies, and heuristics. They used Conditional Random Fields and Support Vector Machines to learn classifiers that make bunsetus that can be considered conditions apart from the others. Unfortunately, their proposal was only evaluated on a small dataset with 3 155 hotel review sentences that was not published.

2.3. Discussion

We have carefully reviewed the literature and we have found out that current entity-relation extractors and opinion miners do not pay any attention to conditions. Neither text-mining frameworks, toolkits, or commercial services take them into account. The few existing proposals to implement condition miners can be roughly classified into two groups, namely: proposals that are based on hand-crafted patterns and a machine-learning proposal. The former are not appealing because there are many unusual ways to introduce conditions, which makes handcrafting patterns with high recall very difficult; furthermore, it is not straightforward to adapt them to other languages in which common connectives are multi-word or there is not a unique, context-agnostic translation for some English connectives. (The previous claims are confirmed in the experimental analysis section.) The only existing machine-learning proposal is tightly bound to the Japanese language, it requires many specific-purpose resources, and the best F_1 score that it attained was 0.58 on a small dataset with 3 155 hotel review sentences.

The conclusion is that mining conditions is an important problem and that it is attracting some researchers. Unfortunately, the few existing techniques have many drawbacks that hinder their general applicability and the lack of a publicly-available dataset makes it difficult to evaluate and compare them from an empirical point of view. This motivated us to work on a new approach that overcomes their weaknesses and outperforms them. Furthermore, we present a new publicly available dataset of conditions.

3. Preliminaries

Preliminary 1 (*Conditionals*)**.** A conditional sentence, or conditional for short, is a sentence with a condition and a consequent; the former describes a state, a factor, or a circumstance that must hold so that the latter holds.

Example 1. Sentence "on entering the hotel, we were delighted to find a spacious welcoming area" is a conditional sentence. Condition "on entering the hotel" is the circumstance that must have hold so that consequent "we were delighted to find a spacious welcoming area" holds.

Preliminary 2 (*Usual Conditionals*)**.** Usual conditionals are expressed by means of grammatical patterns that rely on connectives and verb tenses that are well-known in the literature. The patterns make a difference amongst zero-conditionals, which convey general truths, first conditionals, which convey possible conditions and their likely results, second conditionals, which convey hypothetical conditions and their likely results, and third conditionals, which convey unreal past conditions and their likely results.

Example 2. For instance, "if the picture is blurred, it's as much use as a chocolate teapot" is a zero conditional sentence; "If there's a part of the picture that doesn't change much, you will eventually end up with a ghostly image" is a first conditional sentence; "I would prefer it if they got rid of it" is a second conditional sentence; and "we would have called and asked them a couple of days before arrival if we could have checked in early and they had advised us" is a third conditional sentence.

Preliminary 3 (*Unusual Conditionals*)**.** Unusual conditionals do not fit the patterns that characterise the previous types of conditionals. There is not a standard set of connectives or verb tenses to introduce their conditions.

Example 3. For instance, in sentence "there is a small chat box on the top left of the screen while playing", the condition "while playing" describes a situation in which "there is a small chat box on the top left of the screen" holds. In sentence "the music in Blood Money was excellent especially in levels like 'A House of Cards'", condition "especially in levels like 'A House of Cards'" expresses a set of levels in which the user really likes the music.

Preliminary 4 (*Deep Learning*)**.** Deep learning revolves around a number of machine-learning methods whose focus is on learning feature-based data representations of the input data that facilitate learning classifiers or regressors (LeCun et al., 2015). They typically build on non-linear transformations that are organised in layers so that the outputs of a layer constitute the inputs of the succeeding one. Many deep learning approaches build on neural networks. They have achieved relevant results in computer vision (Szegedy et al., 2017; Tang et al., 2017) and natural language processing (NLP) (Sutskever et al., 2014; Zhai et al., 2017). In the case of NLP, it is necessary to transform the input text into sequences of vectors using so-called word embedders (Mikolov et al., 2013).

Preliminary 5 (*Deep Neural Networks*)**.** A deep neural network is an artificial neural network with multiple hidden layers. Recurrent neural networks (RNN) are very appropriate in NLP. An RNN is a neural network in which the connections between its units form a directed graph across a sequence (Han et al., 2017), which makes them particularly well-suited to deal with sequences of data in which each element depends on the previous ones. Bi-directional recurrent neural networks (BiRNN) (Su and Kuo, 2019) are a particular class of recurrent neural networks that can take both the past and the future elements of a sequence into account. Unfortunately, they both suffer from the so-called exploding and vanishing gradient problems (Pascanu et al., 2013), which can be addressed by controlling the data that is passed on to the next training epoch by means of long-short-term-memory units (LSTM) or gated recurrent units (GRUs) (Su and Kuo, 2019).

4. Mining conditions

The method to mine conditions relies on the deep neural network in Fig. 1. It consists of four layers, namely: a word embedder that transforms the input sentence into vectors that represent its words; an encoder that transforms the embedding-based representation of the input sentence into a context vector that summarises it; a decoder that transforms the context vector into a sequence of IOB tags that determine whether a word belongs or not to a condition; and a builder that extracts the conditions from the input sentence. Additional details are provided below.

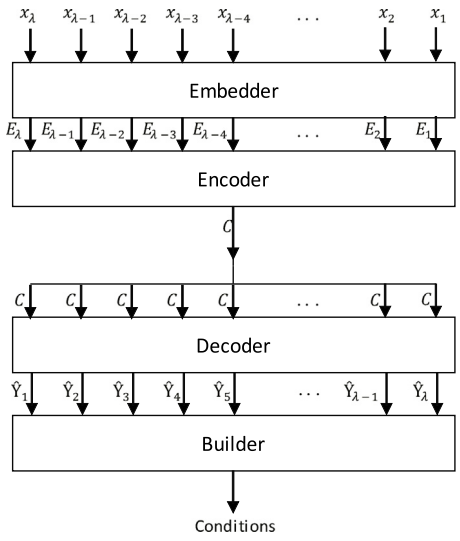


Fig. 1. Sketch of the proposed method.

The embedder. The input are sentences that are encoded as vectors of the form

$$(x_\lambda, x_{\lambda-1}, \dots, x_1), \tag{1}$$

where each x_i represents the corresponding lower-cased, stemmed word in the sentence, and

$$i \in [1, \lambda], \lambda \geq 1. \tag{2}$$

Parameter λ must be set a priori to a large enough length and padding must be used when analysing shorter sentences. The input vectors are fed into an embedder that transforms each word into its corresponding embedding E_i, which is assumed to preserve some similarity to the vectors that correspond to semantically similar words ($E_i \in \mathbb{R}^t$, where t denotes the dimensionality of the word embedding).

Note that the input vectors encode the reversed input sentences because Sutskever et al. (2014) suggested that this approach introduces many short dependencies in the data that make the optimisation problem much easier. To improve the efficiency further, numbers, e-mail addresses, URLs, and words whose frequency is equal to or smaller than five are replaced by class words "NUMBER", "EMAIL", "URL", and "UNK", respectively.

The encoder. Fig. 2 illustrates the two architectures used to implement the encoder, namely: a recurrent neural network (RNN) and a bi-directional recurrent neural network (BiRNN).

They were used because they are particularly well-suited to dealing with natural language because of their ability to process varying-length inputs (even if they must be encoded using fixed-sized vectors with padding). To prevent the exploding and vanishing gradient problems, GRU units are used since they are more efficient than LSTM units because they do not have a separate memory cell. The alternative that consists of an RNN with GRU units is referred to as the GRU encoder and the alternative that consists of a BiRNN with GRU units is referred to as the BiGRU encoder.

The encoder returns a context vector C that captures global features of the input sentences. In the case of the GRU encoder, it is computed as the output of the last GRU unit, that is, $C \in \mathbb{R}^t$; in the case of the BiGRU encoder, it is computed from the last right-to-left GRU unit and the last left-to-right GRU unit, that is,

$$C \in \mathbb{R}^{2t}. \tag{3}$$

The decoder. Fig. 3 illustrates the two architectures used to implement the decoder, namely: one that consists of recurrent neural networks with gated recurrent units (GRU) and another that consists of bi-directional recurrent neural networks with gated recurrent units (BiGRU).

The decoder feeds the context vector into each of the units of the first layer. Then, it computes an output vector D from each recurrent unit of the last layer. Since the number of recurrent units for each layer is λ, the components of the output vector D indicate whether the corresponding input word belongs to a condition or not using the well-known IOB tags, namely: I, which means that a word is inside a condition, O, which means that it is outside a condition, and B, which means that it is the beginning of a condition. The individual components of the output vector are then passed onto a collection of perceptrons that compute the output of the system as follows

$$\hat{Y}_i = \varphi(W D_i + b), \tag{4}$$

where φ is an activation function, W is a weight matrix, D_i is the output of the decoder, and b is a bias vector. $\hat{Y}_i$ represents the probability distribution of the IOB tags as 3-dimensional vectors. The activation function φ is implemented by means of either the Softmax function or the Sigmoid function, since the preliminary experiments that were carried out proved that other choices resulted in worse results.

The builder. To reconstruct the conditions from the output produced by the decoder, it is necessary to take the tag with the highest probability and then return all of the sub-sequences of words in the original sentence that start with a word with tag B that is followed by one, two, or more words with tag I.

Example 4. Fig. 4 shows sentence "I would buy PSP1 when the battery will be improved" as an input to the encoder–decoder model. First, it converts the input words into their lower-cased stems. (The symbol "\0" is used to represent a padding that is ignored across the entire network). Next, it computes their word embeddings (1). Now, the encoder network produces a vector C. It is then used as an input for every first-layer unit of the decoder (3). Finally, the decoder returns the most likely IOB tag for each word. In this example, the condition is then "when the battery will be improved".

5. Experimental analysis

This section describes the computing machinery, the dataset used for evaluation, and the baselines and the alternatives compared; next, the empirical results and the corresponding statistical analysis are presented.

5.1. Computing machinery

The proposed method[2] was implemented with Python 3.5.4 and the following components: Snowball 1.2.1 to stem words, Gensim 2.3.0 to compute word embeddings, and Keras 2.0.8 with Theano 1.0.0 for training the neural networks. The experiments were run on a virtual computer that was equipped with one Intel Xeon E5-2690 core at 2.60 GHz, 2 GiB of RAM, and an nVidia Tesla K10 GPU accelerator with 2 GK-104 GPUs at 745 MHz with 3.5 GiB of RAM each; the operating system used was CentOS Linux 7.3.

5.2. The evaluation dataset

The evaluation dataset[3] is first described; then it is analysed how condition connectives are distributed; finally, it is analysed how similar the conditions are.[4]

[2] The prototype is available at https://github.com/FernanOrtega/encoder-decoder.

[3] The dataset is available at https://www.kaggle.com/fogallego/reviews-with-conditions.

[4] The analyses were performed with the Kaggle kernel that is available at https://www.kaggle.com/fogallego/dataset-analysis.

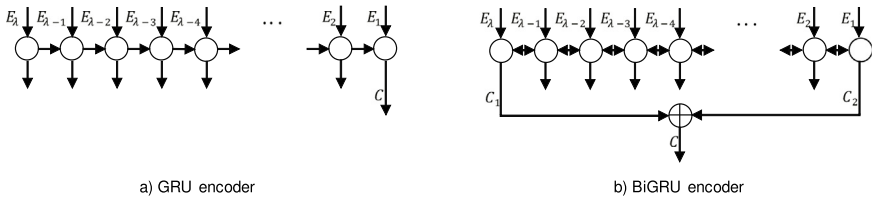


Fig. 2. Architectures of the encoder.

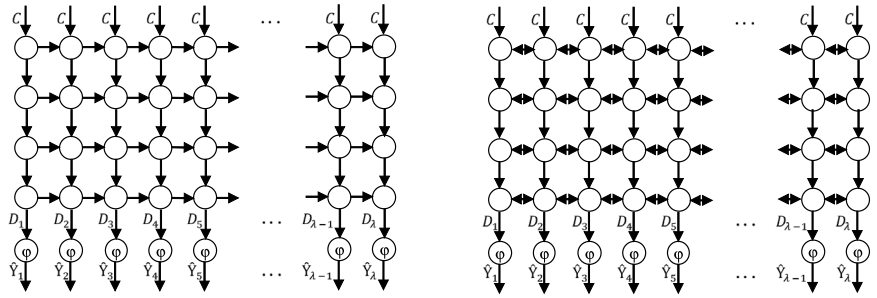
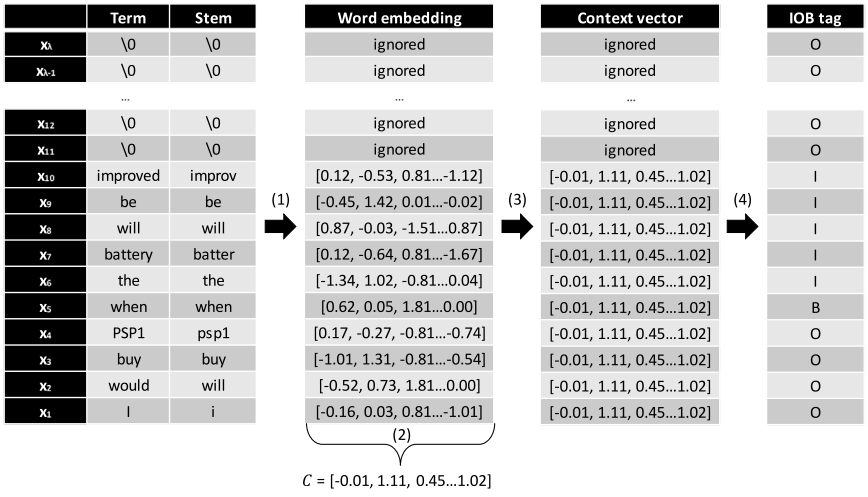


Fig. 3. Architectures of the decoder.



	Term	Stem	Word embedding	Context vector	IOB tag
x_λ	\0	\0	ignored	ignored	O
$x_{\lambda-1}$	\0	\0	ignored	ignored	O
...			...	...	
x_{12}	\0	\0	ignored	ignored	O
x_{11}	\0	\0	ignored	ignored	O
x_{10}	improved	improv	[0.12, -0.53, 0.81...-1.12]	[-0.01, 1.11, 0.45...1.02]	I
x_9	be	be	[-0.45, 1.42, 0.01...-0.02]	[-0.01, 1.11, 0.45...1.02]	I
x_8	will	will	[0.87, -0.03, -1.51...0.87]	[-0.01, 1.11, 0.45...1.02]	I
x_7	battery	batter	[0.12, -0.64, 0.81...-1.67]	[-0.01, 1.11, 0.45...1.02]	I
x_6	the	the	[-1.34, 1.02, -0.81...0.04]	[-0.01, 1.11, 0.45...1.02]	I
x_5	when	when	[0.62, 0.05, 1.81...0.00]	[-0.01, 1.11, 0.45...1.02]	B
x_4	PSP1	psp1	[0.17, -0.27, -0.81...-0.74]	[-0.01, 1.11, 0.45...1.02]	O
x_3	buy	buy	[-1.01, 1.31, -0.81...-0.54]	[-0.01, 1.11, 0.45...1.02]	O
x_2	would	will	[-0.52, 0.73, 1.81...0.00]	[-0.01, 1.11, 0.45...1.02]	O
x_1	I	i	[-0.16, 0.03, 0.81...-1.01]	[-0.01, 1.11, 0.45...1.02]	O

Fig. 4. Encoder–decoder example.

Description. As of the time of writing this article, no dataset to evaluate condition miners was publicly available. So a new dataset was compiled. It consists of 4 671 533 sentences in English, Spanish, French, and Italian that were gathered from Ciao.com between April 2017 and May 2017. The sentences were classified into 16 topics according to their sources, namely: adults, baby care, beauty, books, cameras, computers, films, headsets, hotels, music, ovens, pets, phones, TV sets, and video games.

An application to label the dataset[5] was developed. There is an administrator role that can create labelling tasks and assign them to specific labellers. A labeller performs the task as follows: for each sentence, he or she must spot its conditions, if any; for each condition, he or she must highlight the piece of text that contains the connective.

Table 1 provides a summary of the dataset. The columns of the table denote the language (Lang), the domain (Domain), the number of conditions found (#Conds), the number of sentences (#Sents), the number of sentences that were labelled as of the time of writing this article (#Lab), the number of sentences that contain at least one condition (#SwC), and the corresponding percentage (%SwC).

In Fig. 5, a box and whisker plot that represents the number of words per condition and sentence in the dataset is presented. The length of the input sentences was set to $\lambda = 100$ since this threshold is enough for deal with the vast majority of sentences in the dataset. Note that this threshold does not miss any conditions, but some extremely long sentences that can be considered outliers.

Connective distribution. Table 2 shows the frequency of the five most frequent connectives and the five around the 75-th percentile. Note that there are a few usual connectives that have high frequencies, whereas the others have frequencies that are very low.

The previous table makes it intuitively clear that the distribution of connectives might be a long-tail distribution. To confirm this, it is necessary to compare it to the Power-Law and the Log-Normal distributions, which are the standard long-tail distributions, and to the Exponential distribution, which is not long-tail by definition (Chierichetti et al., 2017; Singh and Tripathi, 2016).

[5] The application is available at http://conditionslabelling.fernanortega.es. You can log in as "labellerTest"/"M1entr45".

Table 1
Summary of the evaluation dataset.

English					
Domain	#Conds	#Sents	#Lab	#SwC	%SwC
adults	325	2 164	1 007	290	28.80%
babycare	26	28 282	1 008	22	2.18%
beauty	19	109 348	1 007	18	1.79%
books	3	49 653	1 002	3	0.30%
cameras	10	8 574	1 010	10	0.99%
films	3	45 547	1 024	3	0.29%
hotels	231	21 908	1 002	214	21.36%
music	113	24 218	1 000	106	10.60%
pets	284	34 878	1 002	251	25.05%
phones	172	2 052	1 027	162	15.77%
tvsets	175	3 537	1 007	159	15.79%
video games	176	111 439	1 006	159	15.81%

Spanish					
Domain	#Conds	#Sents	#Lab	#SwC	%SwC
books	597	1 013 746	2 005	502	25.04%
computers	359	102 021	1 006	303	30.12%
films	210	209 810	1 004	185	18.43%
headsets	243	10 222	1 022	195	19.08%
hotels	173	317 177	1 028	162	15.76%
music	114	417 456	1 033	100	9.68%
oven	177	17 197	1 011	152	15.03%
pets	130	66 845	1 017	114	11.21%
phones	146	547 921	1 010	132	13.07%
tvsets	132	108 287	1 005	108	10.75%
video games	158	537 921	1 008	140	13.89%

French					
Domain	#Conds	#Sents	#Lab	#SwC	%SwC
adults	42	7 586	1 012	42	4.15%
babycare	16	47 972	1 023	16	1.56%
beauty	32	74 739	1 004	30	2.99%
books	42	94 141	1 002	40	3.99%
cameras	61	3 641	1 008	54	5.36%
computers	59	2 792	1 006	56	5.57%
films	59	136 068	1 010	54	5.35%
hotels	46	12 941	1 000	46	4.60%
music	23	25 109	1 004	21	2.09%
pets	49	13 991	1 003	45	4.49%
phones	62	7 972	1 001	61	6.09%
tvsets	39	6 410	1 009	37	3.67%
video games	66	77 457	1 007	61	6.06%

Italian					
Domain	#Conds	#Sents	#Lab	#SwC	%SwC
adults	40	5 068	464	40	8.62%
books	5	5 890	554	5	0.90%
computers	2	3 403	430	2	0.47%
films	2	267 454	1 236	2	0.16%
headsets	139	4 703	1 020	129	12.65%
hotels	37	35 915	895	35	3.91%
music	77	46 029	1 170	68	5.81%
oven	48	2 049	1 041	44	4.23%

Table 2
Connective samples.

Lang	Connective	Freq
en	if	349
	when	239
	for	100
	after	100
	before	64
	while	60

(a.1) Top.

Lang	Connective	Freq
en	every time	2
	over	2
	regardless of	2
	regardless	2
	prior	2
	but in	2

(a.2) 75-th percentile.

Lang	Connective	Freq
es	si	600
	cuando	235
	para	221
	en	119
	al	105
	a	94

(b.1) Top.

Lang	Connective	Freq
es	a medida que	3
	además de	3
	porque si	3
	incluso cuando	3
	a causa de	3
	viendo	3

(b.2) 75-th percentile.

Lang	Connective	Freq
fr	si	384
	même si	63
	pour	48
	mais si	15
	en cas de	11
	quand	9

(c.1) Top.

Lang	Connective	Freq
fr	sauf si	2
	avec	2
	même s'	2
	si bien qu'	1
	si on ne	1
	quant	1

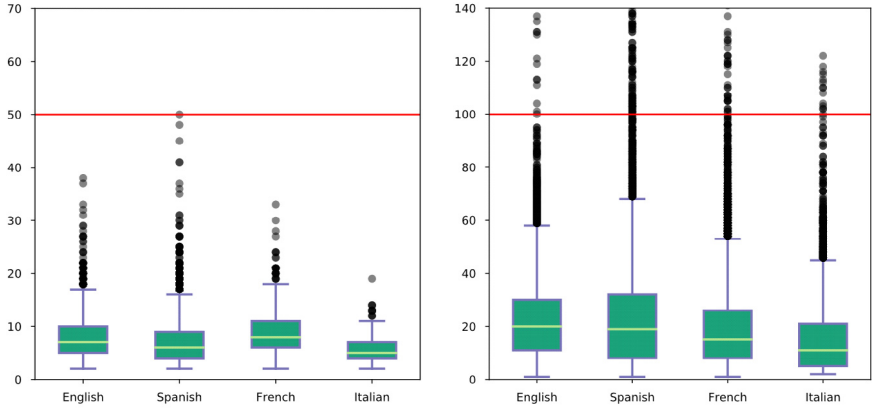
(c.2) 75-th percentile.

Lang	Connective	Freq
it	se	170
	quando	108
	per	7
	anche	7
	in cui	4
	da quando	3

(d.1) Top.

Lang	Connective	Freq
it	el momento in cu	2
	in	2
	in cerca	2
	da	2
	soprattutto	2
	ogni volta	2

(d.2) 75-th percentile.



(a) Per condition. (b) Per sentence.

Fig. 5. Typical numbers of words.

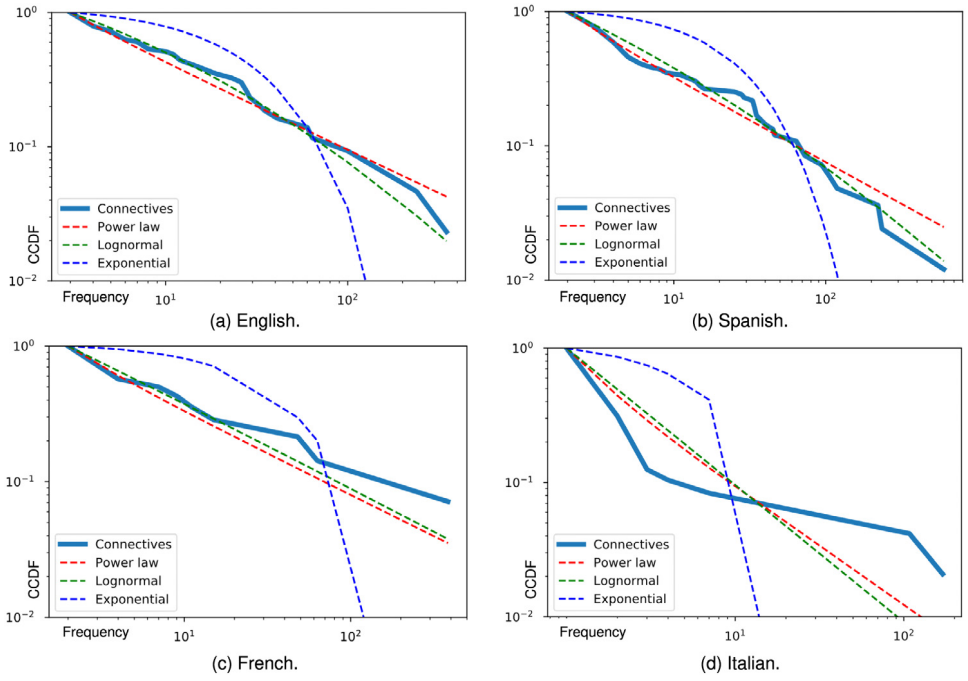


Fig. 6. Connective distribution.

Table 3
Fitting the connective distribution.

Lang	$Dist_1$	$Dist_2$	R	p-value
en	Power-Law	Log-Normal	-1.14	0.33
	Power-Law	Exponential	19.32	0.04
	Log-Normal	Exponential	20.47	0.01
es	Power-Law	Log-Normal	-0.77	0.31
	Power-Law	Exponential	64.32	0.00
	Log-Normal	Exponential	65.09	0.00
fr	Power-Law	Log-Normal	-0.06	0.46
	Power-Law	Exponential	15.40	0.01
	Log-Normal	Exponential	15.46	0.01
it	Power-Law	Log-Normal	-7.00	0.11
	Power-Law	Exponential	70.02	0.00
	Log-Normal	Exponential	77.02	0.00

In Fig. 6, the Complementary Cumulative Distribution Functions (CCDF) of the previous distributions is plotted. Realise that the Power-Law distribution and the Log-Normal distribution are very similar to the connective distribution, whereas the Exponential one is not. A Likelihood Ratio Test (Shafiq et al., 2017) is conducted to check the previous idea statistically. The results of the test are shown in Table 3: for each language in the dataset, the distribution of connectives is compared to every two pairs of the previous standard distributions and the log likelihood ratio R is computed as well as the corresponding p-value.

Independently from the language, the comparison to the Power-Law distribution and the Log-Normal distribution returns p-values that are greater than the standard significance level $\alpha = 0.05$, which indicates that there is not enough empirical evidence to conclude that the connective distribution is significantly different from a Power-Law or a Log-Normal distribution; note that the comparisons to the Power-Law and the Exponential distribution or Log-Normal and the Exponential distribution return a positive log likelihood ratio with a p-value that is smaller than the standard significance level, which indicates that there is enough empirical evidence to conclude that the connective distribution is similar to the Power-Law or the Log-Normal distributions, but very different from the Exponential distribution.

The conclusion is that there is enough statistical evidence to consider the connective distribution a long-tail distribution. Simply put, relying on a collection of handcrafted patterns will typically fall short in terms of recall because there are too many ways to introduce conditions, which clearly argues for a machine-learning solution.

Condition similarity. The similarity of the conditions in the dataset is now analysed. The goal is to find groups of conditions that are similar enough to be modelled using some common features, e.g., verbs, adverbs, or prepositions. To carry this analysis out, every word was changed into lower case and then a vectorisation of each condition was computed as follows: each component of the vectors corresponds to a different word and represents its tf-idf frequency in the condition being vectorised. The English vectorisation has 2 311 words, the Spanish vectorisation has 3 796 words, the French vectorisation has 1 386 words, and the Italian vectorisation has 658 words.

In order to visualise them, a dimensionality reduction was performed by means of two well-known techniques, namely: Isomap and truncated single value decomposition (TSVD). Furthermore, the Gaussian Kernel Density Estimation is computed to better visualise the density of samples with Scott's Rule to compute the estimator bandwidth. In Figs. 7 and 8, a graphical representation of the Isomap and the TSVD projections of the conditions, respectively, is shown. The hues range from bright yellow, which represents the highest densities (conditions that are very similar to each other), to dark blue, which represents the lowest densities (conditions that are not similar to each other). Note that the conditions are organised as follows: there is one small group with high density, a larger group with average density, and a very large group with low density.

As a conclusion, it must not be difficult for a person to learn a rule to mine instances of the first group since there are many examples available and they seem very similar to each other; but it must not be that easy to deal with the many other conditions since they are not similar to each other. This also argues for a machine-learning solution.

5.3. Baselines and alternatives

The methods by Chikersal et al. (2015) and Mausam et al. (2012) were used as baselines. The method by Nakayama and Fujii (2015) was not taken into account because an implementation was not found; neither is it clear how it can be customised to deal with languages other than Japanese.

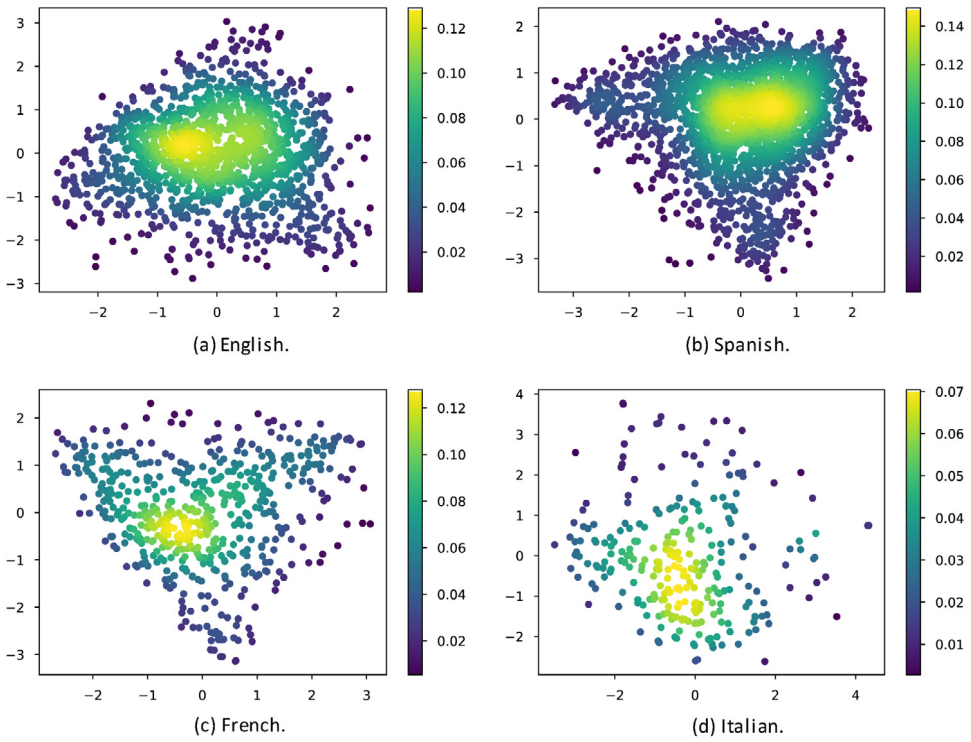


Fig. 7. Condition similarity: Isomap projections. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

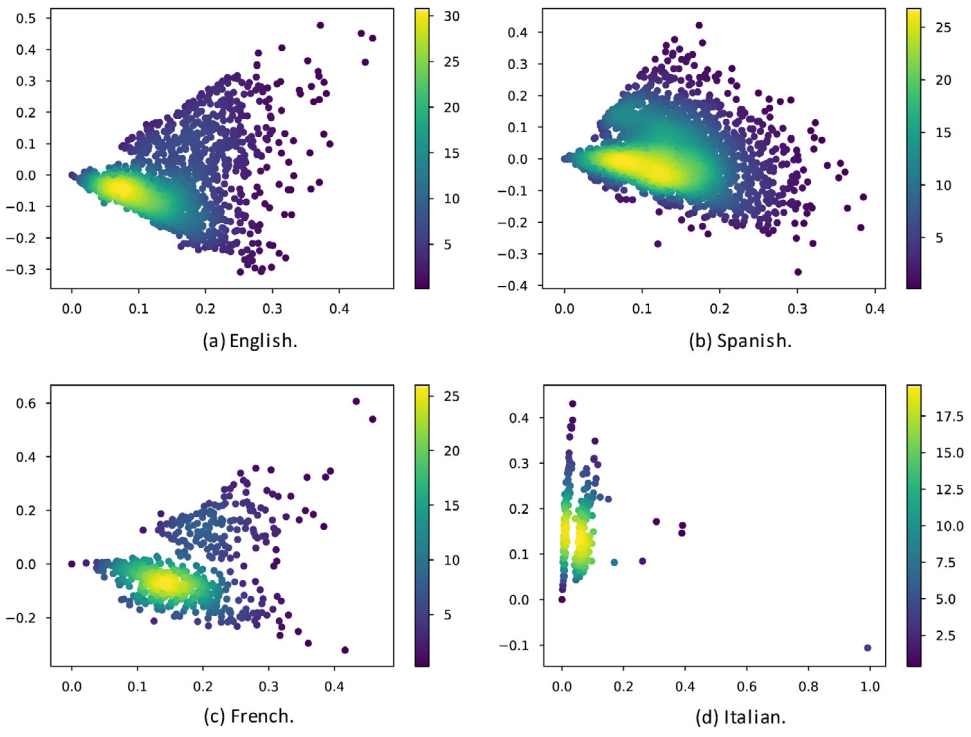


Fig. 8. Condition similarity: TSVD projections. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Regarding the proposed method, eight alternatives were evaluated. They result from combining the two alternatives to implement the encoder (GRU or BiGRU), with the two alternatives to implement the decoder (GRU or BiGRU), and the two alternatives to implement the activation functions in the last layer (Softmax and Sigmoid). Categorical Cross-Entropy is used as the loss function, which is defined as:

$$L(Y, \hat{Y}) = -\frac{1}{\lambda} \sum_{i=1}^{\lambda} \sum_{j=1}^{t} Y_{i,j} \log(\hat{Y}_{i,j}), \tag{5}$$

where Y represents the expected output vector for a given sentence, $\hat{Y}$ represents the output computed by the proposed method, λ represents

Proposal	English			Spanish			French			Italian		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Mausam et al.	0.63	0.61	0.62	0.67	0.53	0.59	0.65	0.67	0.66	0.49	0.50	0.49
Chikersal et al.	0.80	0.46	0.59	0.80	0.44	0.57	0.88	0.58	0.70	0.50	0.48	0.49

(a) Baselines.

Alternatives	English			Spanish			French			Italian		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
GRU-GRU_{sig}	0.65	0.62	0.63	0.61	0.56	0.59	0.87	0.57	0.69	0.58	0.59	0.59
GRU-GRU_{soft}	0.68	0.61	0.64	0.63	0.57	0.60	0.81	0.77	0.79	0.65	0.62	0.63
GRU-$BiGRU_{sig}$	0.65	0.63	0.64	0.67	0.57	0.62	0.82	0.77	0.79	0.63	0.57	0.60
GRU-$BiGRU_{soft}$	0.68	0.62	0.65	0.64	0.58	0.61	0.81	0.79	0.80	0.61	0.60	0.61
BiGRU-GRU_{sig}	0.63	0.61	0.62	0.60	0.56	0.58	0.86	0.65	0.74	0.91	0.50	0.64
BiGRU-GRU_{soft}	0.65	0.62	0.63	0.65	0.56	0.60	0.80	0.78	0.79	0.65	0.59	0.62
BiGRU-$BiGRU_{sig}$	0.67	0.62	0.64	0.63	0.59	0.61	0.82	0.77	0.80	0.60	0.57	0.58
BiGRU-$BiGRU_{soft}$	0.69	0.59	0.64	0.64	0.58	0.61	0.81	0.77	0.79	0.61	0.57	0.59

(b) Encoder-decoder alternatives.

Fig. 9. Experimental results.

#	Proposal	Comparison	z	p-value
1	GRU-BiGRU	GRU-BiGRU x GRU-BiGRU	-	-
2	BiGRU-BiGRU	GRU-BiGRU x BiGRU-BiGRU	0.6606	0.5089
3	GRU-GRU	GRU-BiGRU x GRU-GRU	1.3212	0.1864
4	BiGRU-GRU	GRU-BiGRU x BiGRU-GRU	1.7340	0.0829

(a) Analysis of Softmax-based alternatives

#	Proposal	Comparison	z	p-value
1	GRU-BiGRU	GRU-BiGRU x GRU-BiGRU	-	-
2	BiGRU-BiGRU	GRU-BiGRU x BiGRU-BiGRU	0.3303	0.7412
3	GRU-GRU	GRU-BiGRU x GRU-GRU	2.4772	0.0132
4	BiGRU-GRU	GRU-BiGRU x BiGRU-GRU	3.3029	0.0010

(b) Comparison of Sigmoid-based alternatives

#	Proposal	Comparison	z	p-value
1	GRU-$BiGRU_{Sigmoid}$	GRU-$BiGRU_{Sigmoid}$ x GRU-$GRU_{Sigmoid}$	-	-
2	GRU-$BiGRU_{Softmax}$	GRU-$BiGRU_{Sigmoid}$ x GRU-$BiGRU_{Softmax}$	0.0826	0.9342
3	MB	GRU-$BiGRU_{Sigmoid}$ x MB	2.2295	0.0258
4	CB	GRU-$BiGRU_{Sigmoid}$ x CB	4.1286	0.0000

(c) Comparison of the winning proposal to the baselines

Fig. 10. Statistical analysis.

the size of the input vector, and t represents the size of the word embedding vectors. Furthermore, drop-out regularisations and early stopping when the loss did not improve significatively after 10 epochs are used to prevent over-fitting. The Adam method with batch size 32 is used as the optimiser.

5.4. Experimental results

The proposed method and the baselines were evaluated using 4-fold cross-validation. The standard performance measures were computed, namely: precision, recall, and the F_1 score.

Fig. 9 presents the results of the experiments, where MB and CB refer to Mausam et al.'s and Chikersal et al.'s baselines, respectively. The approaches that beat the best baseline are highlighted in grey.

Although the baselines are naive approaches to the problem, they attain relatively good precision; Mausam et al.'s method attains a recall that is similar to its precision, but Chikersal et al.'s method falls short regarding recall. None of the proposed approaches beat the baselines regarding precision, but most of them beat them regarding recall since they learn more complex patterns thanks to the deep learning approach that projects the input sentences onto a rich feature space that captures many relationships amongst words that are very difficult to spot for a person. Note that the improvement regarding recall is enough for the F_1 score to improve the baselines.

Regarding the activation function, note that precision is better for most of the alternatives when the Softmax activation function is used, but all of the alternatives attain similar results regarding recall independently from the activation function. Finally, the best alternatives for English are GRU-BiGRU when using Softmax and BiGRU-BiGRU when using Sigmoid; the best alternatives for Spanish are BiGRU-BiGRU when using Softmax and GRU-BiGRU when using Sigmoid; the best alternatives for French are GRU-BiGRU when using Softmax and BiGRU-BiGRU when using Sigmoid; and the best alternatives for Italian are GRU-GRU when using Softmax and BiGRU-GRU when using Sigmoid.

5.5. Statistical analysis

To make a decision regarding which of the alternatives performs the best, a stratified strategy that builds on Hommel's test was used. Fig. 10 reports on the results of the statistical analysis. It shows the rank of each approach, and then the comparisons between the best one and the others; for every comparison, the value of the z statistic and its corresponding adjusted p-value are shown.

In the case of the Softmax activation function, the experimental results do not provide any evidences that the best-ranked alternative is different from the others since the adjusted p-value is greater than the standard significance level $\alpha = 0.05$ in every case. In the case of the Sigmoid activation function, the experimental results do not provide any evidences that the best-ranked alternative is different from the second one since the adjusted p-value is greater than the standard significance level; however, there is enough evidence to prove that it is different from the remaining ones since the adjusted p-value is smaller than the significance level. As a conclusion, the GRU-BiGRU alternative is selected in both cases.

Next, the best alternatives are compared to the baselines. The results of the comparison are shown in Fig. 10.c, where the subindex denotes the activation function used. According to the statistical test, there is not enough evidence in the experimental data to make a difference between the best alternatives of the proposed method, but there is enough evidence to prove that it is significantly better than the baselines.

6. Conclusions

In this article, the need for mining conditions in a data engineering context has been motivated and a novel approach to the problem has been presented. It relies on a encoder–decoder model as a means to overcome the drawbacks that were found in the other methods in the literature, namely: it does not rely on user-defined patterns, it does not require any specific-purpose dictionaries, taxonomies, or heuristics, it can mine conditions in both factual and opinion sentences, and it only needs a stemmer and a word embedder, which are readily-available components for many languages. A comprehensive experimental analysis has been performed on a large dataset with 4.7M sentences on 16 common topics in English, Spanish, French, and Italian. The results confirm that the proposed method is similar to the state-of-the-art methods in terms of precision, but it improves recall enough to beat them in terms of the F_1 score. The previous conclusions have been backed up using sound statistical tests. Even through our proposal can be easily adapted to new languages, it depends on obtaining these kind of datasets. It would be interesting to deep dive into so-called cross-lingual approaches than allow us to avoid this limitation.

CRediT authorship contribution statement

Fernando O. Gallego: Methodology, Software, Validation, Investigation, Data curation, Writing - original draft, Writing - review & editing, Visualization. **Rafael Corchuelo:** Conceptualization, Formal analysis, Data curation, Writing - review & editing, Supervision.

Acknowledgements

This work was supported by Opileak.es and the Spanish R&D programme (grants TIN2013-40848-R and TIN2013-40848-R). The computing facilities were provided by the Andalusian Scientific Computing Centre (CICA).

References

Bank, M., Schierle, M., 2012. A survey of text mining architectures and the UIMA standard. In: LREC. pp. 3479–3486, URL http://www.lrec-conf.org/proceedings/lrec2012/summaries/183.html.

Bird, S., Klein, E., Loper, E., 2009. Natural Language Processing with Python. NLTK.

Chierichetti, F., Kumar, R., Pang, B., 2017. On the power laws of language: word frequency distributions. In: SIGIR. pp. 385–394. http://dx.doi.org/10.1145/3077136.3080821.

Chikersal, P., Poria, S., Cambria, E., Gelbukh, A.F., Siong, C.E., 2015. Modelling public sentiment in Twitter. In: CICLing. pp. 49–65. http://dx.doi.org/10.1007/978-3-319-18117-2_4.

Cunningham, H., Tablan, V., Roberts, A., Bontcheva, K., 2013. Getting more out of biomedical documents with GATE's full lifecycle open source text analytics. PLoS Comput. Biol. 9 (2), http://dx.doi.org/10.1371/journal.pcbi.1002854.

Ducange, P., Fazzolari, M., Petrocchi, M., Vecchio, M., 2019. An effective decision support system for social media listening based on cross-source sentiment analysis models. Eng. Appl. Artif. Intell. 78, 71–85. http://dx.doi.org/10.1016/j.engappai.2018.10.014.

Etzioni, O., Fader, A., Christensen, J., Soderland, S., Mausam, 2011. Open information extraction: The second generation. In: IJCAI. pp. 3–10. http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-012.

Han, H.-G., Zhang, S., Qiao, J.-F., 2017. An adaptive growing and pruning algorithm for designing recurrent neural network. Neurocomputing 242, 51–62. http://dx.doi.org/10.1016/j.neucom.2017.02.038.

Honnibal, M., Montani, I., 2018. sPaCy: Industrial-strength natural language processing. Available at https://spacy.io/.

Jin, J., Ji, P., Gu, R., 2016a. Identifying comparative customer requirements from product online reviews for competitor analysis. Eng. Appl. Artif. Intell. 49, 61–73. http://dx.doi.org/10.1016/j.engappai.2015.12.005.

Jin, J., Ji, P., Kwong, C.K., 2016b. What makes consumers unsatisfied with your products: Review analysis at a fine-grained level. Eng. Appl. Artif. Intell. 47, 38–48. http://dx.doi.org/10.1016/j.engappai.2015.05.006.

LeCun, Y., Bengio, Y., Hinton, G.E., 2015. Deep learning. Nature 521 (7553), 436–444. http://dx.doi.org/10.1038/nature14539.

Lima, R., Espinasse, B., Freitas, F., 2019. A logic-based relational learning approach to relation extraction: The OntoILPER system. Eng. Appl. Artif. Intell. 78, 142–157. http://dx.doi.org/10.1016/j.engappai.2018.11.001.

Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D., 2014. The Stanford CoreNLP natural language processing toolkit. In: ACL (System Demonstrations). pp. 55–60. http://dx.doi.org/10.3115/v1/P14-5010.

Mausam, 2016. Open information extraction systems and downstream applications. In: IJCAI. pp. 4074–4077.

Mausam, Schmitz, M., Soderland, S., Bart, R., Etzioni, O., 2012. Open language learning for information extraction. In: EMNLP-CoNLL. pp. 523–534, URL http://www.aclweb.org/anthology/D12-1048.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J., 2013. Distributed representations of words and phrases and their compositionality. In: NIPS. pp. 3111–3119, URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.

Mitchell, T.M., Cohen, W.W., Hruschka, E.R., Talukdar, P.P., Betteridge, J., Carlson, A., Mishra, B.D., Gardner, M., Kisiel, B., Krishnamurthy, J., Lao, N., Mazaitis, K., Mohamed, T., Nakashole, N., Platanios, E.A., Ritter, A., Samadi, M., Settles, B., Wang, R.C., Wijaya, D.T., Gupta, A., Chen, X., Saparov, A., Greaves, M., Welling, J., 2015. Never-ending learning. In: AAAI. pp. 2302–2310, URL http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/10049.

Nakayama, Y., Fujii, A., 2015. Extracting condition-opinion relations toward fine-grained opinion mining. In: EMNLP. pp. 622–631, URL http://www.emnlp2015.org/proceedings/EMNLP/pdf/EMNLP074.pdf.

Narayanan, R., Liu, B., Choudhary, A.N., 2009. Sentiment analysis of conditional sentences. In: EMNLP. pp. 180–189, URL http://www.aclweb.org/anthology/D09-1019.

Pablos, A.G., Cuadros, M., Rigau, G., 2018. W2VLDA: Almost unsupervised system for aspect-based sentiment analysis. Expert Syst. Appl. 91, 127–137. http://dx.doi.org/10.1016/j.eswa.2017.08.049.

Padró, L., Stanilovsky, E., 2012. FreeLing 3.0: Towards wider multilinguality. In: LREC. pp. 2473–2479, URL http://www.lrec-conf.org/proceedings/lrec2012/summaries/430.html.

Pascanu, R., Mikolov, T., Bengio, Y., 2013. On the difficulty of training recurrent neural networks. In: ICML. pp. 1310–1318, URL http://jmlr.org/proceedings/papers/v28/pascanu13.html.

Perikos, I., Hatzilygeroudis, I., 2016. Recognizing emotions in text using ensemble of classifiers. Eng. Appl. Artif. Intell. 51, 191–201. http://dx.doi.org/10.1016/j.engappai.2016.01.012.

Shafiq, M., Atif, M., Viertl, R., 2017. Generalized likelihood ratio test and Cox's F-test based on fuzzy lifetime data. Int. J. Intell. Syst. 32 (1), 3–16. http://dx.doi.org/10.1002/int.21825.

Singh, S., Tripathi, Y.M., 2016. Bayesian estimation and prediction for a hybrid censored Log-Normal distribution. IEEE Trans. Reliab. 65 (2), 782–795. http://dx.doi.org/10.1109/TR.2015.2494370.

Skeppstedt, M., Schamp-Bjerede, T., Sahlgren, M., Paradis, C., Kerren, A., 2015. Detecting speculations, contrasts and conditionals in consumer reviews. In: WASSA@EMNLP. pp. 162–168, URL http://aclweb.org/anthology/W/W15/W15-2923.pdf.

Su, Y., Kuo, C.-C.J., 2019. On extended long short-term memory and dependent bidirectional recurrent neural network. Neurocomputing 356, 151–161. http://dx.doi.org/10.1016/j.neucom.2019.04.044.

Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence-to-sequence learning with neural networks. In: NIPS. pp. 3104–3112, URL https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A., 2017. Inception-V4, Inception-ResNet, and the impact of residual connections on learning. In: AAAI. pp. 4278–4284, URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14806.

Tang, P., Wang, H., Kwong, S., 2017. GoogleNet based multi-stage feature fusion of deep CNN for scene recognition. Neurocomputing 225, 188–197. http://dx.doi.org/10.1016/j.neucom.2016.11.023.

The Apache Software Foundation, 2018. Apache OpenNLP. Available at https://opennlp.apache.org.

Velásquez, J.D., Dujovne, L.E., L'Huillier, G., 2011. Extracting significant website key objects: A Semantic-Web mining approach. Eng. Appl. Artif. Intell. 24 (8), 1532–1541. http://dx.doi.org/10.1016/j.engappai.2011.02.001.

Vicient, C., Sánchez, D., Moreno, A., 2013. An automatic approach for ontology-based feature extraction from heterogeneous textual resources. Eng. Appl. Artif. Intell. 26 (3), 1092–1106. http://dx.doi.org/10.1016/j.engappai.2012.08.002.

Yang, S.-Y., Soo, V.-W., 2012. Extract conceptual graphs from plain texts in patent claims. Eng. Appl. Artif. Intell. 25 (4), 874–887. http://dx.doi.org/10.1016/j.engappai.2011.11.006.

Yoo, S., Song, J., Jeong, O., 2018. Social media contents based sentiment analysis and prediction system. Expert Syst. Appl. 105, 102–111. http://dx.doi.org/10.1016/j.eswa.2018.03.055.

Zhai, F., Potdar, S., Xiang, B., Zhou, B., 2017. Neural models for sequence chunking. In: AAAI. pp. 3365–3371, URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14776.

Zhang, H., Sekhari, A., Ouzrout, Y., Bouras, A., 2016. Jointly identifying opinion mining elements and fuzzy measurement of opinion intensity to analyze product features. Eng. Appl. Artif. Intell. 47, 122–139. http://dx.doi.org/10.1016/j.engappai.2015.06.007.