Named Entity Recognition in Tweets

Introduction

The Named Entity Recognition (NER) is a sub-field of information extraction. The purpose is to extract and classify some entities from a text. Some actual Named Entity Recognition systems have really high results, such as the MUC-7 model which excels with a 93.39% score[1], not far from the human annotators performance.

An example of extraction would be:
[Steve Jobs]*PER*, [Steve Wozniak]*PER* and [Ronald Wayne]*PER* created [Apple]*ORG* in [Los Altos]*LOC* in [California]*LOC*.
Where *PER* stands for Person, *ORG* for organization and *LOC* for location.

The corpus for which the system has been trained matters a lot. The performance of standard NLP tools can be significantly diminished on other corpus. In particular, the tweets 140-character limit and informality can cause significant problems to the pre-existing NER models. In this project, we have tried implementing a classifier trained particularly on tweets. We have used the help of others NLP tools to improve its quality significantly.

The Stanford NER system has been used for the training and the testing of our classifier, using the 2 and 10 cross-validation evaluation. We have learned on three following entity types only : Person, Organization and Location.

[1] Elaine Marsh, Dennis Perzanowski, "MUC-7 Evaluation of IE Technology: Overview of Results", 29 April 1998

First Classifier

A CoNLL file annotated with the three named entities was available to train on. We have used the Stanford NER to build a model and evaluate it. We have used the basic properties given as example on austen.prop on the stanford CRF FAQ https://nlp.stanford.edu/software/crf-faq.shtml#a. The results are the following:

	Precision	**Recall**	**F1**
K=2	0.743	0.350	0.474
K=10	0.792	0.515	0.623

On-going evaluation

All of the improvement have been first trained, for efficiency reason, on the first half and tested on the second half. Even though the evaluation results are less accurate this way, it allowed us to test more options, as the evaluation time was considerably shorter. The purpose was to determine the features and properties that seemed to improve the classifier, so that at the end we could try to evaluate on a complete cross-validation.

We are aware that it could create a bias on the results of a property, but due to the high number of different properties we have tried, it was the most reasonable way to get results.

On this basis, the evaluation of the implemented classifier is:

Precision	**Recall**	**F1**
0.7308	0.368	0.4895

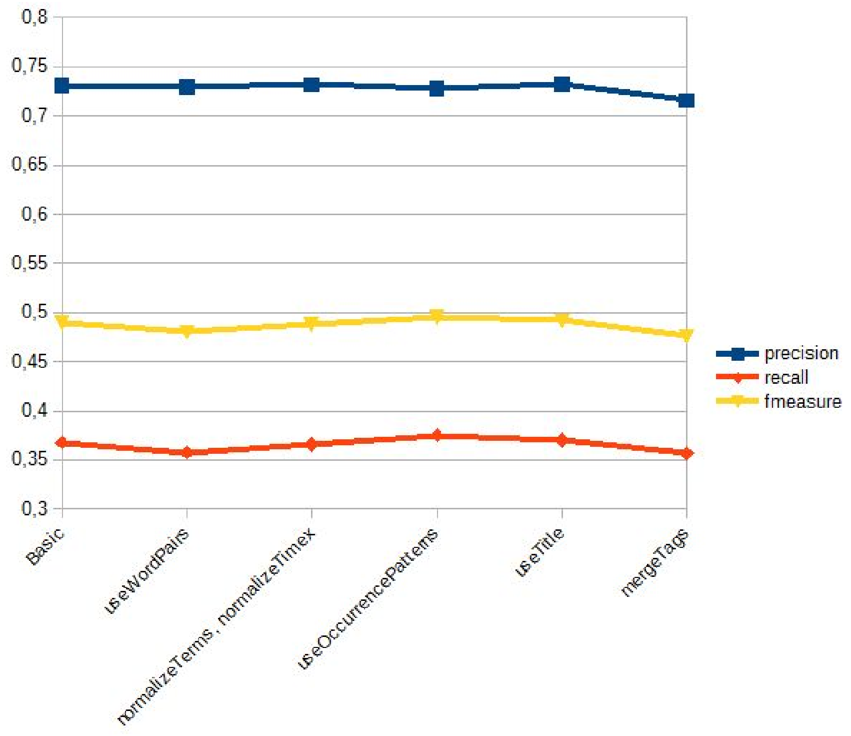
As we've used different libraries for adding new features during the project, and to automatize the process of creating the right CoNLL file, we've built a script from the basic raw tweets file to the CoNLL file with the good columns and features.

Properties

On the first improvement, we have tried modifying or adding new properties to the .prop file. The list as well as the description of almost all properties can be found there: https://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/ie/NERFeatureFactory.html.

The following tables as well as the graph synthesize the results.

	Precision	**Recall**	**F1**
Basic	0.7308	0.368	0.4895
useWordPairs	0.7296	0.3577	0.4801
normalizeTerms, normalizeTimex	0.7316	0.3662	0.4881
useOccurrencePatterns	0.7277	0.3753	0.4952
useTitle	0.7321	0.3705	0.492
mergeTags	0.7157	0.3565	0.476



We realize than even though some of the properties may have a positive impact on precision and/or recall, the variation is very subtle. It seems however that mergeTags is having a negative impact on the results, as well as use useWordPairs. Moreover, useOccurencePatterns and useTitle are the most promising ones.

A considerable list of other properties have also been tried, without having any change on the results, as we may not have used them in the right combination or they just don’t apply to our case

Gazette

A gazette file is a list (or dictionary) of entities. In our case, a gazette file can contain first names, locations, organizations and so on. In the NER field, the use of gazette files can overcome the lack of an extended list of entity names in the training file. It is used by the system as an additional information to train on.

By using gazette files for PER, ORG, LOC and MISC, we expected significant better results, especially for the recall, as we would provide known popular entities to the classifier. We started by using the gazette incrementally.

Our first gazette was a list of 5 000 common first names, at first in uppercase then in capital case. Then, by observing the entity the classifier had missed or misclassified, we decided to extend the gazettes to a file of locations: states and capital city of the US, as well as the countries all around the world. We also added big organizations, in terms of capital size, as well as famous corporations name. And finally we completed with some MISC entities, such as popular movies, championships and events.

For minimizing the error risk, the entities are all in capital case only.

Another important feature of the gazette, is the choice between sloppyGazette and cleanGazette. The former means that any of the words on the line can be matched as an entity, whereas the latter considers that the entire line entry is one match.

At first, the format we used for the first names was :

PER James Jones Robert Michael…

In this case, the sloppyGazette property was used. But it turned out to complicate the case of locations for example, where United States would be considered as two entities instead of one. We then changed our gazette format to the following:

LOC United States

LOC United Kingdom

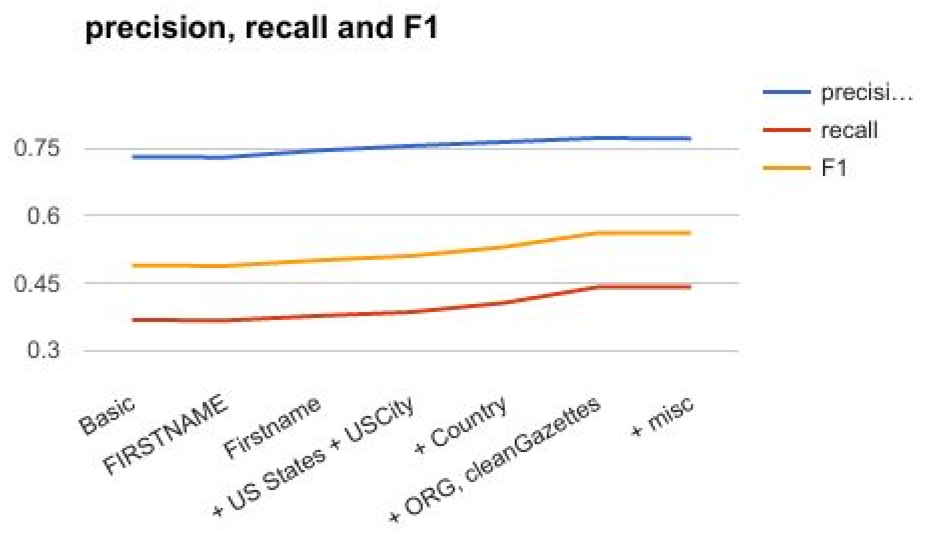
LOC ...

The cleanGazette property should, in this case, be used.

The results of the progressive improvements we have made are presented in the following table:

	Precision	**Recall**	**F1**
Without Gazette	0.7308	0.368	0.4895
FIRSTNAME	0.7301	0.3668	0.4883
Firstname	0.7452	0.3771	0.5008
Firstname + US States + USCity	0.7556	0.3856	0.5106
Firstname + US States + USCity + Country	0.7642	0.4062	0.5304

Firstname + US States + USCity + Country + ORG, cleanGazettes	0.7731	0.4413	0.5618
Firstname + US States + USCity + Country + ORG, + misc, cleanGazettes	0.7714	0.4413	0.5614



The improvement were significant, in terms of precision and recall, changing the F1-measure from 0.4895 to 0.5618.

Even though the misc gazette didn't improve the results we decided to keep it, as it did not significantly altered them either.

Part-of-Speech Tags

A valuable feature commonly used in NER are the Part-of-Speech tags. The process of tagging a sentence is to mark a word in a text as a particular category, subcategory of nouns, verbs, adjectives, etc. An example of POS tagging of a tweet would be:
*'The/**NN** truth/**NN** will/**MD** always/**RB** win/**VB**' - Julian/**NNP** Assange/**NNP** writes/**VBZ** #FakeHashtag/**HT**.*

On the Stanford NER faq page, they don't think it is necessary to use one, as the features used by the NER system are very similar to their POS tagger. Moreover, the informality of the tweets could have complicated the POS tagging, resulting in many mistaggings that could have worsen the classification.

However in the paper about Named Entity Recognition in Tweets[2], they have improved greatly their classifier by using an adapted POS tagging, so we decided to try adding this new feature.

We have used the default Stanford tagger, as well as two other POS tagger that are tuned especially for tweets. The first one is GATE Twitter tagger[3]. They take into consideration the common abbreviations on twitter, slang, common misspellings and etc.The CMU Noah's ARK tweet tagger[4] is the second one.

We tagged the original tweet file with the different taggers and merged it with the CoNLL file, so that the classifier could learn on the tags as well.

The results are the following.

	Precision	**Recall**	**F1**
Basic	0.7308	0.368	0.4895
useTags (stanford)	0.7138	0.4062	0.5177
useTags (tweetie-tagger)	0.6701	0.4389	0.5304
useTags (ark-tagger)	0.6645	0.4352	0.526

[2] Ritter, A., Clark, S., & Etzioni, O. (2011, July). Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 1524-1534). Association for Computational Linguistics.
[3] https://gate.ac.uk/wiki/twitter-postagger.html
[4] http://www.cs.cmu.edu/~ark/TweetNLP/

It turns out the the tweetie POS tagger gave the better F1 results, by greatly increasing the recall. However, all the POS tagger, while considerably increasing the recall, also decreases the precision a lot. Depending on the purposes of the NER, it might be better to preserve the precision of the results instead of the recall. However, without a defined purpose, we want the best classifier overall and decide to focus on the F1-measure.

Lemma

Lemmatization is useful to normalize the different version of a same word to only one version, the one we can find in the dictionary. For example, “I’ve been studying my notes” would become “I have be study I note”.

Lemmatization simplifies the learning process and eliminates inflected forms of words, thereby potentially improving the classifier performance[5].

We decided to use the Stanford CoreNLP Lemmatizer to lemmatize the full file. We merged the results with our annotated CoNLL file and evaluated the classification. We obtained the following results:

	precision	**recall**	**F1**
Basic	0.7308	0.368	0.4895
Lemma	0.7424	0.3717	0.4954

The lemmatization improves the precision, recall, and F1-measure. The results, not as impressive as the ones from the gazette or POS tagging are however satisfying.

5 https://web-beta.archive.org/web/20141212094149/http://brown.cl.uni-heidelberg.de/~sourjiko/NER_Literatur/survey.pdf

Combination

After having used those different features available to train our classifier, our next purpose was to combine them to get the best classification.

	Precision	Recall	F1
Basic	0.7308	0.368	0.4895
Gazette	0.7714	0.4413	0.5614
Tag	0.6701	0.4389	0.5304
Lemma	0.7424	0.3717	0.4954

Tag + Gazette	0.7333	0.4861	0.5846
Tag + Gazette + Lemma	0.7444	0.5061	0.6025

By combining the different features, we increased F1 measure from 0.4895 to 0.6025. As we can see, the tag features greatly decrease the precision. However, combined with the Gazette, and the Lemmatizer, the precision loss is counterbalanced. The recall has been greatly improved, from 0.368 to 0.5061 percents.

Henceforth, the base to compare our next improvements with will be the combined model of Gazette + Lemma + Tag.

We then studied the different properties suggested by the Stanford NER on our combined form. Those are the results we obtained:

	Precision	Recall	F1
Gazette + Lemma + Tag	0.7444	0.5061	0.6025
useTitle	0.7436	0.5091	0.6044
useTitle2	0.7465	0.5097	0.6058
normalizeTerms, normalizeTimex, (useNB)	0.7455	0.5054	0.6025
useWordPairs	0.7448	0.5	0.5983
useTypeSeqs3	0.7453	0.5048	0.6019
useOccurrencePatterns	0.7482	0.5091	0.6059
useOccurrencePatterns, useTitle2	0.7484	0.5097	0.6064

We notice that useTitle2, recommended by the Stanford NER and useOcurrencePatterns increase the precision as well as the recall.

We tried to improve some of the gazette features, by modifying the global tag PER to B-PER for the first name for example, and same for location, organization and misc…

	Precision	Recall	F1
Gazette + Lemma + Tag	0.7444	0.5061	0.6025
change PER to B-PER on firstname Gazette	0.7487	0.5103	0.6069
change LOC, MISC, ORG to I-LOC or B-LOC..., cleanGazette	0.7477	0.5006	0.5997
change LOC, MISC, ORG to I-LOC or B-LOC..., sloppyGazette	0.7464	0.5024	0.6006

It proved to be efficient on the first names only but not on location, organization or misc, as it separated the entities in B-LOC United and I-LOC States instead of keeping them whole.

We finally looked the .prop files of the existing Stanford NER models and decided to add some of the recommended properties. For example, they recommend using wordShape=dan2useLC and disjunctionWidth=4 when using on CoNLL data. We also removed the limitation on maxNGramLeng and used the default value(which is the whole NGrams) as suggested, and set sigma value to 20.

	Precision	Recall	F1
Gazette + Lemma + Tag	0.7444	0.5061	0.6025
useOccurrencePatterns, useTitle2	0.7484	0.5097	0.6064
change PER to B-PER on firstname Gazette	0.7487	0.5103	0.6069
useOccurrencePatterns, useTitle2, PER to B-PER	0.7478	0.5115	0.6075
useOccurrencePatterns, useTitle2, PER to B-PER, recommended properties	0.7442	0.5266	0.6168

The maximum F1 value we obtained on the K=2 first half cross-validation is 0.6168. It is a 0.127 increase compared to the first basic 0.4895.

We would have tried to compare the different recommended values, to know exactly the effect of each, but due to lack of time, we have not done so.

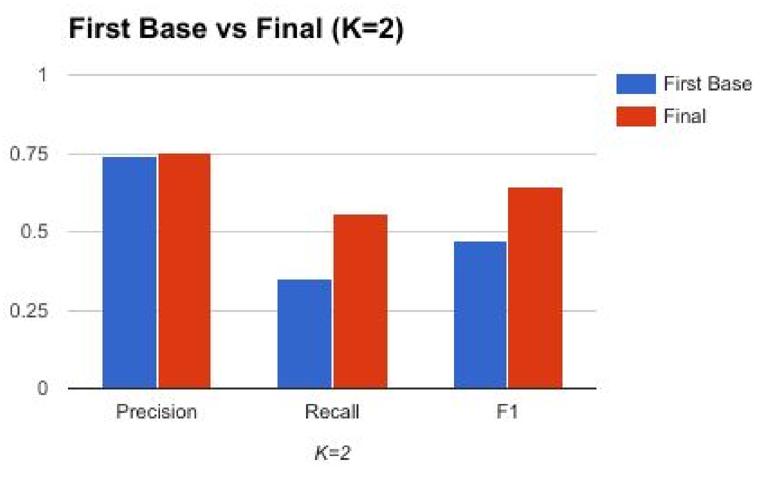
Evaluation

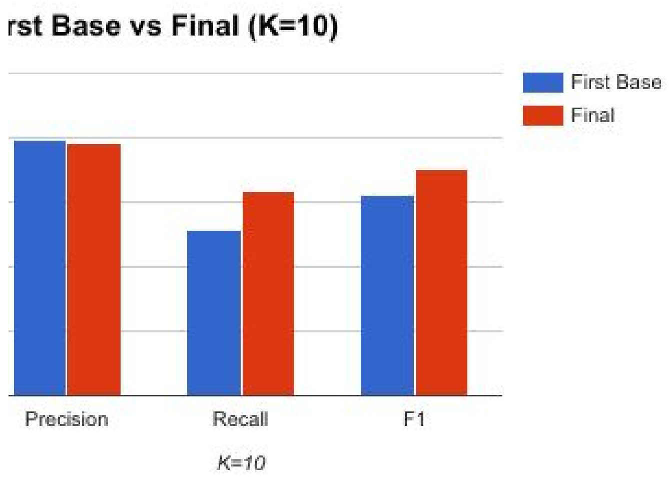
After having decided the optimal classification process, we were ready to evaluate our new system on the cross-validation for K=2 and K=10.

We obtained the following results:

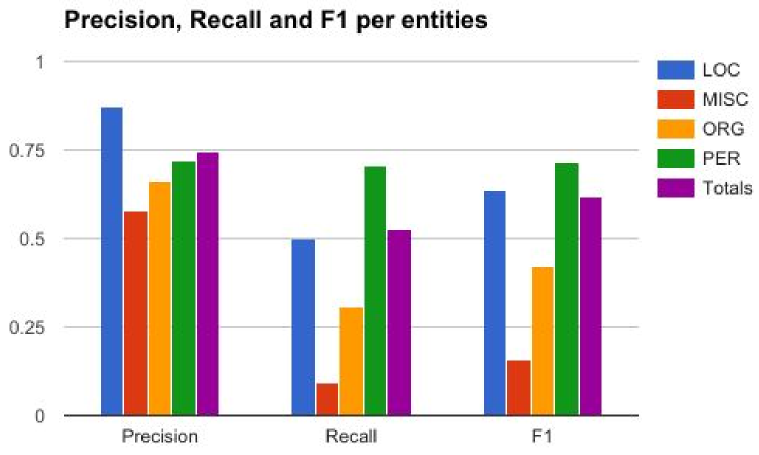
	precision	**recall**	**F1**
K=2	0.7535	0.5597	0.6419
K=10	0.7816	0.6362	0.7005

The following graphs show the comparison between the results on the basic case and the results after the various improvements.





We also compared the performances of the different named entity categories on K=2.



As we can see, the classifier performances is really heterogeneous depending on the entities. Organizations and particularly Miscellaneous entities have the lowest precision and recall, having a really low F1-measure.

Limits and reflexions

The major limits of our projects were the memory constraints and the time limit.

In fact, the stanford NER system requires a lot of memory to train a classifier, thus forcing us to use some properties to limit the memory consumption in order to prevent running out of memory. As well, the entire computer was dedicated at learning, using a lot of resources and making us think twice when trying a new property or combination.

The limitation on time made us have to cease experimenting new models, even though there were still features that looked interesting.

For example, in *Named entity recognition in tweets: an experimental study*[6], they used a chunking tagging to improve their results. The last feature that we would have like to try, also mentioned in this paper was the capitalization feature. In fact, they implemented a capitalization classifier to predict whether a tweet was informatively capitalized or not. This information would have been really useful as the NER classification relied a lot on the capitalization, but the tweets are often wrongly capitalized. It would have informed the classifier on the pertinence of the capitalization information.

On top of that, the lack of time did not allow us to evaluate every property modification or feature on K=2 or K=10 cross-validation, which would have given more accurate values, maybe resulting in a better global K=10 evaluation. As we can see, the K=2 increase is better than the K=10. Indeed, the precision that we thought we had increased on K=2, proved to be decreased on K=10.

We also want to point out that the choices have been made in order to increase the global F1 value, probably at the expenses of the precision. It also gave more importance to PER classification than MISC because there are more PER named entities. This might partially explain the low results for this category.

We also could have combined the results with the existing Stanford CRF models, which could have increased the recall and the precision. However, we thought that it would have been out of the scope of this project.

[6] Ritter, A., Clark, S., & Etzioni, O. (2011, July). Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 1524-1534). Association for Computational Linguistics.