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 accuracy 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.