The University of Texas System Submission for the Code-Switching Workshop Shared Task 2018

Florian Janke, Tongrui Li, Gualberto Guzmán,
Eric Rincón, Barbara Bullock, Almeida Jacqueline Toribio
Bilingual Annotation Task (BATs) Research Group
University of Texas at Austin
{florian,tli1998,eric.rincon,gualbertoguzman}@utexas.edu
{bbullock,toribio}@austin.utexas.edu

Abstract

This paper describes the system for the Named Entity Recognition Shared Task of the Third Workshop on Computational Approaches to Linguistic Code-Switching (CALCS) submitted by the Bilingual Annotations Tasks (BATs) research group of the University of Texas. Our system uses several features to train a Conditional Random Field (CRF) model for classifying input words as Named Entities (NEs) using the Inside-Outside-Beginning (IOB) tagging scheme. We participated in the Modern Standard Arabic-Egyptian Arabic (MSA-EGY) and English-Spanish (ENG-SPA) tasks, achieving weighted average F-scores of 65.62 and 54.16 respectively. We also describe the performance of a deep neural network (NN) trained on a subset of the CRF features, which did not surpass CRF performance.

1 Introduction & Prior Approaches

Named entity recognition (NER) and classification are essential tasks in information extraction (Nadeau and Sekine, 2007). However, NER in texts in which multiple languages are represented is not straightforward because NEs can be language-specific (e.g., *Estados Unidos* in Spanish vs. *United States*) or language-neutral but regionally specific (e.g., *Los Angeles*) or even mixed (e.g., *Nueva York* in Spanish) (Çetinoglu, 2016; Guzman et al., 2016). The task is further complicated by the fact that names of companies, institutions and brands in one language can be common nouns in another (e.g, *Toro* is a brand name for a U.S. company but *toro* in Spanish means bull'). These challenges confound the already difficult task of working with multilingual texts, which can be considered resource scarce' with respect to the availability of NLP tools (Riaz, 2010; Zirikly and Diab, 2015; Sitaram and Black, 2016; Guzmán et al., 2017). But NER in multilingual communication is essential given that multilingualism is common throughout the world, and, for many speakers, language mixing is a shared practice and one that can be prevalent in social media like Twitter (Jurgens et al., 2014; Jamatia et al., 2015, 2016; Vilares et al., 2015).

2 Data Description

Over 62k Tweets were collected and manually annotated for NEs to be used in this shared task (Aguilar et al., 2018). The annotators labeled each NE using one of ten tags: PERSON, LOCATION, ORGANIZATION, PRODUCT, GROUP, EVENT, TIME, TITLE, OTHER, or NOT-NE. All tokens are tagged using the IOB scheme while ignoring hashtags and @-mentions, i.e. *Louis Vuitton* is tagged with B-ORG and I-ORG but *@RideAlong* is tagged as O. NEs can occur in all languages and, since this is Twitter data, can frequently be misspelled or missing orthographic features that would ease identification. The Tweets were divided into training, development, and test sets and released to the participants of the shared task along with tools for preprocessing of the Tweets.

3 Approach & Methodology

3.1 Conditional Random Field

One approach we used to perform NE recognition in this shared task was the usage of conditional random fields (Lafferty et al., 2001), a technique used for sequence labeling. More specifically, *python-crfsuite* (Peng and Korobov, 2014) was used, a Python wrapper around *CRFsuite* (Okazaki, 2007), an implementation of CRFs in C/C++. CRFs work by looking at several words

Proceedings of The Third Workshop on Computational Approaches to Code-Switching, pages 120–125
Melbourne, Australia, July 19, 2018.

and their features and expected classification (in this case the NE classification) as examples and using the information gained to predict classifications on future data that has not been seen before. For our use of *CRFsuite*, the values of 1.0 for L1 and 0.001 for L2 regularization (from the NER example provided by the package) were used with a total of 150 training iterations. All other parameters were left at their default values.

3.1.1 Features Used

Several different features of the tweets as whole and individual tokens were used as input, some of which rely on external resources to generate. Initially we developed our features on the ENG-SPA dataset. Interestingly many of the features used for ENG-SPA performed well on the MSA-EGY data. Inspiration for the features used was drawn from various papers from the *First Workshop on CALCS* (Chittaranjan et al., 2014; Lin et al., 2014). The features used can be grouped into five categories:

1. Word features: lowercase copy of the word, its two last characters, length, whether it is the first word or not, whether this word is all alphanumeric characters (only for the MSA-EGY dataset), if this word is made up of digits or not, and if the word contains emoji.

2. Capitalization: is the word all uppercase or title case?

3. Language tags: off-the-shelf taggers from the *Natural Language Toolkit* (NLTK) (Bird and Loper, 2004) were used to perform NE and part of speech (POS) tagging on one tweet at a time and the tags were applied to individual tokens.

4. Language detection: in the ENG-SPA dataset only, language detection on entire tweets was done using langdetect, a Python port (Danilák, 2017) of language-detection (Nakatani, 2010) originally written in Java. Probabilities of the tweet being English or Spanish rounded to 2 digits after the decimal point were used. If the tweet was classified as neither English or Spanish, the probability was set to be 0. For example, "Quiero un roadtrip asap" was falsely classified as Romanian.

5. Twitter functionality: does the overall tweet contain an @mention or #hashtag? Is this word itself one of the two? Is this a URL?

A subset of the features mentioned above were applied to the next and previous words and used as features to classify the current word: the word in lowercase form, its last two characters, if it is the first word, title case, uppercase, a URL, @-mention, or #hashtag, if it contains an emoji, its NE and POS tag classification by NTLK.

Additional features have been experimented with and their results are included in section 4. These features include the last three characters of the word, whether it contains a digit (not if it is a digit itself), or if it is made up of exclusively ASCII characters.

3.2 Deep / Wide Model

The deep and wide architectures have had recent success for the use of recommendation engines (Cheng et al., 2016), but here we adapt it for the use of NER. Deep and wide architectures have the benefit of embedding categorical variables in a vector space allowing for unseen feature combinations and the use of cross-product feature transformations for effective and interpretable features. This combination of cross-product feature combinations and dense embeddings allows for deep and wide models to memorize and generalize to the input data while reducing feature engineering efforts.

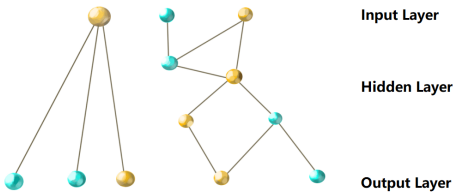

Figure 1: layers in wide (left) and deep (right) models

3.2.1 Training process

The model was trained using Tensorflow, an open-source machine learning framework designed by Google (Abadi et al., 2016). The classifier provides a general purpose wide and deep learning model for users to train. The wide model is a pre-built linear classifier which attempts to classify each word in a particular tweet based on values from their linear combinations.

The deep model used a pre-built neural network to classify the data by letting its features

propagate through the network. Using Python, Tweets from the tsv file were first parsed into a internal data model where the features are computed as properties of the individual words. The model outputs a csv file with each feature listed as a column that can be conveniently passed to the `DNNLinearCombinedClassifier`. We used a subset of the CRF features including the word itself, capitalization of the word, word type, and the adjacent words.

The wide portion of the model enables NER tagging through linear properties. Features were inputted as the base column to provide information to the activation layer of the neural network. Some features such as the word, word's capitalization, word's type were cross validated as a set and hence would make the model recognize that these grouped features would have dependencies among themselves. Implementing a neural network, the deep model greatly increased the training time with a ratio of roughly 1:20 per iteration. The models did not perform well against the CRF possibly due to a lack of features, hence the CRF was used in the final submission of the project.

4 Results & Analysis

4.1 CRF Performance

Our submission for the shared task was evaluated using both the harmonic mean F1 and the surface forms F1 metrics (Derczynski et al., 2017) on each dataset. In line with the baseline performance, our system performs better on the MSA-EGY data than the ENG-SPA data despite the difference in data size. The scores on the two challenges were 65.62 for MSA-EGY and 54.16 for ENG-SPA. After the shared task submission closed, we continued experimenting with different features. The F1-scores (computed using scikit-learn (Pedregosa et al., 2011)) of the CRF trained on the training data set and evaluated on the testing set using various configurations of features are shown in table 2. These results are different from those submitted to the competition as they were evaluated on a different data set.

Inclusion or omission of certain features affected the two sets of data differently: for example including the ASCII feature improves scores for ENG-SPA but decreases that for MSA-EGY. The last row (special) shows an attempt to maximize the score by combining successful individual features and while scores do increase, this attempt does not perform as well as expected. For ENG-SPA the submitted configuration excluding POS and NE seems to work best while the submitted configuration with a combination of changes (shown in table 1) works best for MSA-EGY going by F1-score.

Table 1 shows the features that were modified for use. An asterisk (*) indicates that this is a change compared to the submitted configuration *a*. Rows not included are features that remained unchanged throughout.

4.2 NN Performance

As shown in table 3, the F1-score was suboptimal due to a low recall score. Two different models, one implementing only the wide portion and the other implementing the deep and wide models were trained with features extracted from the data set. Three different variants of the features and the results are displayed in table 2. Surprisingly, the wide model showed an overall better performance than the wide and deep model. This may be due to a lack of the features extracted from the dataset for the deep learning to build on. The lack of recall may occur due to the same reason, which eventually leads to the rejection of this model.

5 Conclusion

In this paper, we described the University of Texas BATs research group's submission for the CALCS 2018 Shared Task for NER. We found that some features improved results of the CRF model on one language combination, but not on the other. In both cases, our CRF model outperformed the baseline NER performance. However, training an NN using the same features as the CRF did not significantly improve F1-scores, but further feature engineering on or combination of both models could improve the performance.

	ENG-SPA								MSA-EGY							
Features	a	b	c	d	e	f	g	h	a	b	c	d	e	f	g	h
en prob	✓	✓	*	✓	✓	✓	✓									
es prob	✓	✓	*	✓	✓	✓	✓									
ar prob											✓*					✓*
last 3 chars		✓*						✓*		✓*						✓*
has emoji	✓	✓	✓	✓	*	✓	✓	*	✓	✓	✓	✓	*	✓	✓	*
ascii				✓*				✓*				✓*				
NE	✓	✓	✓	✓	✓	*	✓	*	✓	✓	✓	✓	✓	*	✓	*
POS	✓	✓	✓	✓	✓	*	✓	*	✓	✓	✓	✓	✓	*	✓	*
two words							✓*		✓	✓	✓	✓	✓	✓	*	✓

Table 1: Feature configurations

	ENG-SPA			MSA-EGY		
Configuration	precision	recall	F1-score	precision	recall	F1-score
a (submission)	0.69	0.25	0.32	0.86	0.68	0.76
b (include last 3 characters)	0.67	0.26	0.33	0.84	0.7	0.76
c (toggle language probabilities)	0.49	0.24	0.31	0.86	0.68	0.76
d (check for ascii)	0.73	0.25	0.34	0.86	0.67	0.75
e (no emoji)	0.71	0.25	0.33	0.87	0.68	0.76
f (exclude POS and NE tags)	0.69	0.27	0.34	0.86	0.68	0.76
g (toggle surrounding two words)	0.50	0.23	0.30	0.84	0.65	0.73
h (special)	0.63	0.27	0.33	0.84	0.71	0.77

Table 2: Performance of CRF on various configurations

	ENG-SPA (Wide model only)			ENG-SPA (Deep + Wide model)		
Configuration	precision	recall	F1-score	precision	recall	F1-score
original	0.23	0.03	0.05	0.22	0.0373	0.06
excluding next word	0.28	0.04	0.07	0.13	0.02	0.03
excluding next word and length	0.31	0.04	0.07	0.13	0.03	0.05

Table 3: Performance of NN on various configurations

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.

Gustavo Aguilar, Fahad AlGhamdi, Victor Soto, Mona Diab, Julia Hirschberg, and Thamar Solorio. 2018. Overview of the CALCS 2018 Shared Task: Named Entity Recognition on Code-switched Data. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, Melbourne, Australia. Association for Computational Linguistics.

Steven Bird and Edward Loper. 2004. Nltk: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 31. Association for Computational Linguistics.

Özlem Çetinoglu. 2016. A turkish-german code-switching corpus. In *LREC*.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS 2016, pages 7–10, New York, NY, USA. ACM.

Gokul Chittaranjan, Yogarshi Vyas, Kalika Bali, and Monojit Choudhury. 2014. Word-level language identification using crf: Code-switching shared task report of msr india system. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pages 73–79.

Michal Danilák. 2017. Python port of google language detection library.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147. Association for Computational Linguistics.

Gualberto Guzmán, Joseph Ricard, Jacqueline Serigos, Barbara E Bullock, and Almeida Jacqueline Toribio. 2017. Metrics for modeling code-switching across corpora. *Proc. Interspeech 2017*, pages 67–71.

Gualberto A Guzman, Jacqueline Serigos, Barbara E Bullock, and Almeida Jacqueline Toribio. 2016. Simple tools for exploring variation in code-switching for linguists. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 12–20.

Anupam Jamatia, Björn Gambäck, and Amitava Das. 2015. Part-of-speech tagging for code-mixed english-hindi twitter and facebook chat messages. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 239–248.

Anupam Jamatia, Björn Gambäck, and Amitava Das. 2016. Collecting and annotating indian social media code-mixed corpora. In *the 17th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 3–9.

David Jurgens, Stefan Dimitrov, and Derek Ruths. 2014. Twitter users# codeswitch hashtags!# moltoimportante# wow. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 51–61.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Chu-Cheng Lin, Waleed Ammar, Lori Levin, and Chris Dyer. 2014. The cmu submission for the shared task on language identification in code-switched data. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 80–86.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Shuyo Nakatani. 2010. Language detection library for java.

Naoaki Okazaki. 2007. Crfsuite: a fast implementation of conditional random fields (crfs).

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Terry Peng and Mikhail Korobov. 2014. python-crfsuite. `https://python-crfsuite.readthedocs.org/`.

Kashif Riaz. 2010. Rule-based named entity recognition in urdu. In *Proceedings of the 2010 named entities workshop*, pages 126–135. Association for Computational Linguistics.

Sunayana Sitaram and Alan W Black. 2016. Speech synthesis of code-mixed text. In *LREC*.

David Vilares, Miguel A Alonso, and Carlos Gómez-Rodríguez. 2015. Sentiment analysis on monolingual, multilingual and code-switching twitter corpora. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 2–8.

Ayah Zirikly and Mona Diab. 2015. Named entity recognition for arabic social media. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 176–185.