Location NER from Tweets

Stone Fang (Student ID: 19049045)
Computers and Information Sciences
Auckland University of Technology
Auckland, New Zealand
fnk7060@autuni.ac.nz

I. Metrics

- exact match: both entity type and boundaries are matched
- relaxed match

II. Algorithms

A. Knowledge/Rule-based

hand-crafted rules. rely on lexicon resources and domain specific knowledge.

pros: do not require annotated training data high precision

cons: need domain experts construct and maintain the knowledge resources low recall

B. Unsupervised and bootstrapped systems

earliest systems,

pros: required very minimal training data.

C. Feature-engineered supervised learning

learn models from data, but need manual feature engineering common approaches:

- Hidden Markov Models (HMM)
- Support Vector Machines (SVM)
- Conditional Random Fields (CRF)
- decision trees

D. Feature-inferring neural networks

features are also learned from data, e.g. embeddings

1) Representation(Embedding):

- Word level Architecture
- Character level Architecture
- Character+Word level architectures
- Character + Word + affix model

2) Context encoder:

- CNN
- RNN -> LSTM -> bi-LSTM -> window bi-LSTM
- Recursive neural networks
- Neural Language Models
- Attention -> Transformer -> BERT/GPT/ELMo

3) Tag decoder:

- Multi-Layer Perceptron + Softmax
- Conditional Random Fields
- RNN
- Pointer Networks

III. Algorithm analysis

A. choosed algorithms

This paper conducted a two-stage location extraction system for tweets, where a location predictor followed by an extractor. In the first stage, a classification model predicts if an input text contains any location or not, where only inputs with positive predictions will be feed into next stage. In the second stage, a combined NER model from various tools extracts locations from the text filtered out by previous stage. By a variety of experiments, the author demonstrated that: 1) combining various NER tools can improve F1-score of the model; 2) the final precision of location extraction can be improved if a proper predictor is set before it.

1) Location Extraction: This is the second stage in the whole pipeline but it was studied first. In this stage, a compound model is combined by some of the three NER tools: Ritter tool, the Gate NLP framework (Gate) and the Stanford NER. In addition, in some combinations the result will be post-filtered by DBPedia. To achieve this, each tool is applied to input texts to extract locations, then the individual outputs are merged as final result. To filter the locations, locations extracted from NER models are kept only if they appears in DBPedia.

In the experiment, combined NER extractors are compared to the Ritter tool as the baseline. The results showed that combining with either Gate or Stanford NER could improve the model performance under the evaluation of F1-score. Moreover, the DBPedia post-filtering dramatically increased precision but also caused great decline in recall. The best F1-score on Ritter dataset was obtained by "Ritter tool + Stanford NER + DBPedia", and "Ritter tool + Stanford NER" for MSM2013 dataset.

2) Location Prediction: In location prediction stage, the author first showed the usefulness of prior predictors by passing tweets through a perfect (that is, 100% correct results) location predictor before feeding into extractor, where the precision were significantly improved with the recall unchanged. Though perfect prediction models are impossible in real-world applications, this is a reasonable proof-of-concept. Then several features were defined for predictive models, including:

- **Geography gazetteer**: if any word is included by a geography gazetteer (Gate was chosen in the experiment)
- **Prep**: if input text contains any of the 7 prepositions regarding to place (*at, in, on, from, to, toward, towards*)

- **Prep + PP**: if input text contains a preposition directly followed by a proper noun (PP)
- **Place + PP**: if any word of place (*town, city, state, region, department, country*) appears directly after or before a proper noun
- **Time**: if any word regarding to time exists, including *today, tomorrow, weekend, tonight*, as well as all the names of months and the days of a week
- **DefArt + PP**: if any proper noun appears after the definitive article "the"
- **Htah**: if the tweet contains any hash tag
- **PP, Adj, Verb**: the counts of proper nouns, adjectives and verbs in the text

After features generated, three machine learning algorithms are trained on Ritter and MSM2013 datasets with 10-folds cross validation, namely Naive Baiyes (NB), Support Vector Machine (SVM) and Random Forest (RF). Experiments under various parameters have been carried out.

The author found that some features are more useful than others. Though there are minor difference between performances on two datasets, most significant common features includes: Geography gazetteer, Prep+PP, PP, and Place+PP. On Ritter dataset, the best F1-score was 65% with an accuracy of 94%, achieved by RF under threshold 0.5. On MSM2013 dataset, the best F1-score was 61% along with accuracy 84%, also obtained by RF but with a different threshold 0.75. Naive Bayes is also a comparable solution which is preferred over SVM.

After that, various predictors are applied before extractor for evaluation. In this experiment, only Ritter tool was used instead of combined tools, and RF models with different thresholds were used as predictors. The result showed that the precision was significantly improved, while the recall decreased due to prediction error. Another advantage of predictor is reducing the number of tweets that are fed into extractor.

3) Summary: To summarise, this work showed that 1) combining different NER tools is a reasonable way to improve the location extraction model metrics such as F1-score; 2) a preceding location predictor can benefit the precisions of final result but reduce the recall. Furthermore, from the experiment of perfect predictor conclusions can be drawn that better prediction model will mitigate the recall decline.

IV. Proposed Approach

A. Data Preparation

There are some public available datasets containing texts from social media with human annotations, including Ritter [1] and WNUT [2]. In addition, more data can be gathered by Twitter's API and annotated by crowdsourcing. Moreover, as there are some existing works having already demonstrated that gazetteer features can improve neural NER models [3], [4], we will use the same gazetteers from UIUC NER system as in the experiment of [4].

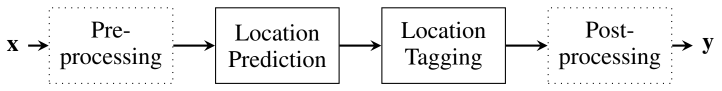

Fig. 1. Proposed Algorithm: Two-Stage Location Extraction

B. Evaluation

NER systems are usually evaluated by recall and precision [3], [5], [6], which are controversial in nature though. A balanced metrics taking into account both sides is called F1-score, also known as F-score or F-measure:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

C. Algorithm Description

Our proposed NER algorithm for locations on Twitter follows the two-stage model as studied by Hoang and Mothe [5], but the machine learning models in each stage will be replaced by deep learning ones which was employed by Mao, Thakur, Sparks, *et al.* [6]. Before and after this model, pre- and post-processing are incorporated for necessary tasks such as POS tagging and formatting. The overall process of our proposal is shown as Fig 1.

1) Pre-processing: Before input text is fed into the two-stage location extraction model, some pre-processing will be beneficial for the subsequent stages. Basically in this phase the input text will be tokenized and Part-of-Speech (POS) will be tagged, because according to some research [5] the POS tag can increase the final model performance. In addition, since tweets are highly informal texts, for example, with many spelling errors, spelling correction will also be carried out as normalisation in this phase.

2) Location Prediction: The work of Hoang and Mothe [5] showed that a good location predictor can increase the metrics of the location extraction model. However, this work also showed that while this preceding filter increased precision, it did have negative impact on recall, which has controversial impact on the final F1-score. This is reasonable because some locations are missing because of false negative prediction. Therefore, the performance of predictor is crucial; a less efficient one would even decrease the overall performance in terms of F1-score.

In [5] three classification model has been tested: Naive Bayes, Support Vector Machine and Random Forest. However, in a more recent study, Lee and Dernoncourt [7] has demonstrated that deep learning models outperform traditional machine learning algorithms on classification of short texts like tweets. Lee and Dernoncourt [7] also showed that generally CNN had a better metrics over RNN, so our location predictor will be a CNN-based text classification model, which consists of three components: embedding, CNN and full-connected(FC). First, in the embedding layer, some popular pre-trained embeddings such as GloVe or BERT will be used. In addition, since Hoang and Mothe [5] has showed that features including gazetteer and POS tags were significant to final results, we will include these features in our model. Specifically, a variable indicating if the word exists in gazetteer and POS tags generated from pre-processing step are concatenated into word embeddings. Second, multi layered CNN model will be followed. Finally the output of CNN will be fed into a full-connected layer for prediction. Because the result is binary, that is, whether the input contains any location or not, softmax is not necessary. Instead, a logistic activation function is sufficient.

Once a probability has been produced by prediction model, a threshold is required for decision-making. Selection of this threshold is more of a trade-off between recall and precision, and will be optimised based on the dataset and application. For example, in some application where precision is preferred over recall, we can increase the threshold for stricter filtering.

3) Location Tagging: After an input tweet passes the filtering by location predictor, a sequence tagging model will be applied for entity tagging. We use the biLSTM-CRF model which has been shown outperformed Standford NER tool [3], [6]. In addition, char-level embeddings is also an effective method to improve the overall performance so it will also be included in our model [3].

First, the input word sequence will be fed to an embedding layer which combines both word embedding and char-level embedding. A pre-trained word embedding called GloVe was adopted by Mao, Thakur, Sparks, *et al.* [6], but some other ones such as BERT are also worth trying. In addition, incorporating extra knowledge such as gazetteers and POS tags will improve the overall system performance [3], therefore, these two features will also be appended to vector representation. Second, the combined embedding vectors are give to a bidirectional LSTM (bi-LSTM) layer. The output of bi-LSTM is transformed by a softmax module into probabilities over all tag categories. Third, the sequences of these probabilities are fed to a CRF layer for maximum likelihood estimation over entire sequence.

Following Mao, Thakur, Sparks, *et al.* [6] and also Stanford NER, the final tags generated by this tagging model is encoded by standard Beginning/Inside/Outside (BIO) format.

4) Post-processing: Since the output of location tagging model is a sequence of labels, the actually locations will be formed by post-processing. All B- and I- tags will be retrieved and formatted into entities.

References

[1] A. Ritter, S. Clark, O. Etzioni, *et al.*, "Named entity recognition in tweets: An experimental study," in *Proceedings of the conference on empirical methods in natural language processing*, Association for Computational Linguistics, 2011, pp. 1524–1534. [Online]. Available: https://dl.acm.org/citation.cfm?id=2145595.

[2] L. Derczynski, E. Nichols, M. van Erp, and N. Limsopatham, "Results of the WNUT2017 shared task on novel and emerging entity recognition," in *Proceedings of the 3rd Workshop on Noisy User-generated Text*, Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 140–147. DOI: 10.18653/v1/W17-4418. (visited on 04/28/2020).

[3] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Transactions on Knowledge and Data Engineering*, 1–1, 2020, ISSN: 2326-3865. DOI: 10.1109/tkde.2020.2981314.

[4] T. Liu, J.-G. Yao, and C.-Y. Lin, "Towards improving neural named entity recognition with gazetteers," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 5301–5307. DOI: 10.18653/v1/P19-1524. (visited on 05/12/2020).

[5] T. B. N. Hoang and J. Mothe, "Location extraction from tweets," *Information Processing & Management*, vol. 54, no. 2, pp. 129–144, Mar. 2018, ISSN: 03064573. DOI: 10.1016/j.ipm.2017.11.001.

[6] H. Mao, G. Thakur, K. Sparks, J. Sanyal, and B. Bhaduri, "Mapping near-real-time power outages from social media," *International Journal of Digital Earth*, vol. 12, no. 11, pp. 1285–1299, 2019. DOI: 10.1080/17538947.2018.1535000. [Online]. Available: https://doi.org/10.1080/17538947.2018.1535000.

[7] J. Y. Lee and F. Dernoncourt, "Sequential short-text classification with recurrent and convolutional neural networks," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 515–520. DOI: 10.18653/v1/N16-1062.