Mining Tweets that refer to TV programs with Deep Neural Networks

Takeshi S. Kobayakawa **Taro Miyazaki** **Hiroki Okamoto** **Simon Clippingdale**
{kobayakawa.t-ko,miyazaki.t-jw,okamoto.h-iw,simon.c-fe}@nhk.or.jp

NHK (Japan Broadcasting Corporation)
1-10-11 Kinuta, Setagaya-ku, Tokyo 157-8510, JAPAN

Abstract

The automatic analysis of expressions of opinion has been well studied in the opinion mining area, but a remaining problem is robustness for user-generated texts. Although consumer-generated texts are valuable since they contain a great number and wide variety of user evaluations, spelling inconsistency and the variety of expressions make analysis difficult. In order to tackle such situations, we applied a model that is reported to handle context in many natural language processing areas, to the problem of extracting references to the opinion target from text. Experiments on tweets that refer to television programs show that the model can extract such references with more than 90% accuracy.

1 Introduction

For some decades, opinion mining has been among the more extensively studied natural language applications, as plenty of consumer-generated texts have become widely available on the Internet. Consumer-generated texts in the real world are not always ”clean” in the sense that vocabulary not in dictionaries is frequently used, so some measures for handling out-of-vocabulary (OOV) words are required. (Turney, 2002) gave a solution to this problem in the form of a semantic orientation measure, defined by pointwise mutual information, to automatically calculate the polarity of words.

However, these kinds of measures, usually called sentiment analysis, are only one aspect of opinion mining; another big problem to be tackled is the detection of the target of the opinion. Unlike analyzing opinions about, say, a well-known product that is referred to by name without many variations, analyzing opinions about an inconcrete object such as media content requires the extraction of the opinion target. Real tweets that refer to television (TV) programs frequently do not explicitly mention the proper full name of the program. Although official hashtags supplied by broadcasters are sometimes used, unofficial hashtags may also appear, and on occasion, paraphrased versions of the content may be used without either hashtags or the program name. Thus some method for finding paraphrases in that context is required in order to extract the target of such tweets.

Following the advent of Deep Neural Networks (DNNs), many context processing models have been proposed. One of the most successful models is Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), which we adopt as the basis for context processing. The recurrent architecture of LSTM is thought to handle long-term dependencies.

Our task is to detect references to TV programs as described in section 3. Viewers of TV programs generate many tweets, and broadcasters pay much attention to what viewers say, including what specific part of a program is being discussed. Producers and directors want to know as specifically as possible what viewers talk about, in order to assess in detail the impact that their programs have on audiences.

Formally, our task is to extract relevant parts from a sentence, which is similar to named entity recognition (NER) in the sense that it is a sequence detection problem, but rather more semantic. Our motivation is to clarify how well various NER models work on our task. The contribution of this paper is the performance comparison, on our task, of three NER methods that are reported to perform at state-of-the-art levels. We also conducted the same experiment on the CoNLL 2003 NER task, to allow comparison against our task.

2 Related Work

Related to our task in this study is the extraction of opinion targets in sentiment analysis that was conducted as a shared task in SemEval 2016,

Proceedings of the 2019 EMNLP Workshop W-NUT: The 5th Workshop on Noisy User-generated Text, pages 126–130

called aspect-based sentiment analysis (Pontiki et al., 2016), where opinion target extraction was one measure of performance for a sentence-level subtask. Unlike other sentiment analysis tasks, such a task requires the extraction of entity types including the opinion target and attribute labels as aspects of the opinion. However, entities to be extracted remain at the word level, and the candidates are given, such as "RESTAURANT", "FOOD", etc. Aspects to be extracted are similar in that one word can be chosen among given candidates, such as "PRICE", "QUALITY" and so on. In our task, the opinion target to be extracted is not restricted to a word but rather can be a phrase, and is not in general specified in advance. There have been many studies related to paraphrases, one of which was a shared task in SemEval 2015, known as paraphrase identification (Xu et al., 2015).

As regards phrase extraction, NER has a long history from (Tjong Kim Sang and De Meulder, 2003). The state-of-the-art models are thought to be (Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016).

3 Task and Data

The task is to extract references to TV programs in the text part of tweets. We call such expressions "referers". Figure 1 shows these notions with an example. The referer part is not always the proper name of the program or an officially-defined hashtag, but can be a paraphrased reference to the program content.

Figure 1: Terminology with an example

The targeted TV program is a Japanese TV drama, described in Table 1. We prepared a population of tweets that refer to TV programs by selecting tweets manually in a best-effort manner: tweets that contain wider general terms are likely to contain some portion of targeted data (including the broadcaster name NHK, for this study) if transmitted during the broadcast time of the program. Tweets were then selected manually to prepare research data.

The referer parts in the text are annotated manually as a region, using the brat rapid annotation tool by (Stenetorp et al., 2012). Since such annotations are performed at the character level before the tokenization process, labels for the sequence tagging problem are converted to the positions of tokens during the tokenization process. The coding scheme for the region of the reference is IOB tags (Ramshaw and Marcus, 1995).

The tweets and targeted program names are both in Japanese, and since Japanese has no spaces between words, a Japanese tokenizer is used to separate words. We used SentencePiece (Kudo, 2018), a kind of subword tokenizer that handles OOVs and de-tokenization well. SentencePiece is trained with the same training data as the main task. Raw data are as described in Table 2. Sequence lengths in terms of words and characters are given as averages and standard deviations. Table 3 shows the characteristics of annotated tags. The referer part is annotated more finely, i.e. subcategorized by type of reference such as people, scene, music, etc., but for this study, we gather them into a single type of reference. Almost one third of the tokens has some kind of reference to the targeted program, and many chunks consist of more than one token, since there are many I-REFERENCE tags in the corpus. The data thus prepared are used for both training and evaluation.

Broadcast time	2019.4.1 8:00-8:15
Broadcaster	NHK (GTV channel)
Program title	Natsuzora[1]
Genre	television drama series
Synopsis	The story of an animator who decides to go to Tokyo.

Table 1: Targeted TV program

# tweets	3,745	
# chars per. tweet	30.1(ave.)	19.5(SD.)
# words per. tweet	11.6(ave.)	9.4(SD.)
# vocab of chars	1,693	
# vocab of words	7,727	

Table 2: Statistics of Raw Data

# B-REFERENCE	7,871
# I-REFERENCE	5,558
# O	29,829

Table 3: Statistics of Reference Tags

[1] https://en.wikipedia.org/wiki/Natsuzora

4 Model and Training Procedure

We treat the extraction of referer sections as a sequence tagging problem, and the state-of-the-art model for such a sequence tagging problem is a LSTM model combined with CRF, as reported in (Huang et al., 2015). We used a modified version of LSTM-CRF [2], implemented by TensorFlow [3].

The models used have three types of layers. Inputs for the model are a sequence of tokenized words, and to deal with large vocabulary tasks, distributed representations are used. The first layer is a trainable embedding layer that inputs sequences of words. The second layer is a recurrent layer, LSTM, where contexts are handled. The third layer is a CRF layer. The Viterbi decoding becomes the model output. For robustness purposes, a dropout (Hinton et al., 2012) layer is inserted at each layer, and can be thought of as a kind of regularizer.

Models are trained to maximize the f1 score (harmonic mean of precision and recall), and training is stopped when there is no further improvement. We tried three variants of these models, details of which are described as follows.

4.1 Bidirectional LSTM-CRF

The basic type of LSTM-CRF model was discussed in (Huang et al., 2015). The model consists generally of three layers: embedding, recurrent, and CRF.

Although several pre-trained models are available for the embedding layer, such as GLoVE (Pennington et al., 2014) or Word2Vec (Mikolov et al., 2013), we elected to train the embedding itself during the training procedure.

For the recurrent layer, contexts are handled by the LSTM type cell, whose input is whole sequence of words (distributed reps.) of a text, and whose output is a sequence of the same length as the input. The input is treated bi-directionally, so that a reversed word order is equivalent, in order to handle both forward and backward context dependencies. Forward and backward computations are performed separately, and they are concatenated just before the next CRF layer.

At the CRF layer, the concatenated outputs from the preceding recurrent layer are input to a linear-chain CRF. Like the original CRF (Lafferty et al., 2001), output labels are also used in the estimation of subsequent outputs.

[2] https://github.com/guillaumegenthial/tf_ner

[3] TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. http://tensorflow.org/

4.2 Character Embeddings

Given the sparsity problem with vocabularies, characters (the components of words) are used and combined with words. Like (Lample et al., 2016), characters are fed into the embedding layer and their parameters are also trained like the word input layer. The embeddings of both words and characters are concatenated, for input to the following recurrent layer.

4.3 Character Convolutions

There is also a model that uses convolutions for character inputs. (Ma and Hovy, 2016) used a convolutional neural network for characters, which then performed max-pooling. We also evaluated this model.

5 Experiments

5.1 Data allocation

Data with referer tags, as described in section 3, were divided into sets for training, validation, and evaluation, in the proportions 90%, 5%, and 5%, respectively.

The three models described in the previous section were compared on two tasks. One task is the original CoNLL 2003 Named Entity Recognition task (Tjong Kim Sang and De Meulder, 2003) in English. Named entities here are persons, locations, organizations, and names of miscellaneous entities, found in the Reuters news corpus. The second task is the task for this study, described in section 3.

We used texts without part-of-speech tags. Details of the training parameters are given in Table 4. Character type parameters are only used for those models that include character-level modeling. The training took 10 to 20 minutes on a laptop computer. Training was stopped at around 4,000 iterations.

Mini-batch size	20
Char. embedding dims.	100
Word embedding dims.	100
Char LSTM size	25
Word LSTM size	100
Dims. of context representations	300
Dropout rate	0.5

Table 4: Training Parameters

5.2 Results

The results are shown in Table 5. Figures for accuracy, precision, and recall have the same meanings

Task	Model	Accuracy(%)	Precision(%)	Recall(%)
CoNLL 2003	Majority Voting	82.54	**100.00**	0.05
	LSTM-CRF	94.15	75.61	69.97
	With char-emb.	95.92	80.35	77.90
	With conv. of char-emb.	**96.19**	81.00	**79.76**
Extraction of referer part for TV program extraction from tweets	Majority Voting	78.12	7.73	5.11
	LSTM-CRF	90.27	76.53	**82.95**
	With char-emb.	**91.23**	**77.38**	82.70
	With conv. of char-emb.	91.06	76.71	**82.95**

Table 5: Results

as in CoNLL 2003. Accuracy is an overall correct ratio including O tags (which means containing no kind of tags of interest). Precision is a measure for extracted instances, while recall relates to relevant instances, as usual in information retrieval parlance. The three models described in section 4 are compared together with majority voting as a trivial baseline model. The trivial model chooses the most frequent output seen in the training data as the trained output. Bold-faced figures are the best results among the four models compared.

Figures for the CoNLL 2003 NER task are almost the same as those given for the state-of-the-art models, so the implementation seems correct. On the CoNLL 2003 NER task, models that use convolution of character embeddings were the best performing, as reported in (Ma and Hovy, 2016). The 100% precision attained by majority voting comes at the price of extremely low recall, so it is not of much use; majority voting works very conservatively, only working when confident of the occurrence.

Figures for our task are original, and first reported here as far as we know. Unlike the NER task, the best performing model except for recall is LSTM-CRF with simple character embeddings, while simple word-level LSTM-CRF with convolutional character embeddings performed best for recall. Convolution of character embeddings performed a little worse than the model without convolutions. This may be due to over-modeling of characters, when in fact they are not so important for this task, while character level modeling remains effective.

6 Discussions

The experiments showed that referer sections for TV programs were well extracted using the state-of-the-art models for sequence tagging. However, the performance on this task was somewhat different than that on the NER task. This is because the extracted parts are longer than named entities, and tend to form explanatory phrase expressions. These expressions can be thought of as phrase-level coreferences, or paraphrases, which are thought to relate linguistically to the high-level understanding of natural languages, such as rhetorical structures.

One possibility is to improve the embedding layer. Several phrase-level embeddings have been studied, and they may be useful for this kind of task. As words and characters are combined, phrases can also be combined to represent input sequences, and such models are probably worth trying.

A second possibility is to improve the recurrent layer. For deeper context handling, simply stacking LSTM layers is proposed. Techniques from semantic parsers may also help in capturing semantic chunks from the whole sentence. Whether further handling of contexts is possible is of much interest.

7 Conclusions and Future Work

We applied sequence tagging models to study the performance of extracting referer sections from relevant tweets for a targeted TV program. The extraction accuracy achieved by LSTM-CRF was significantly better than that attained by majority voting. Further treatment of deep contexts is suggested by comparisons of the experimental results on NER tasks, which remains a topic for future work. We suspect that some variations of deep neural networks may be able to solve this problem, especially for this kind of domain, because although noisy, large amounts of data addressing the same topic are available.

Acknowledgment

The authors would like to acknowledge the significant contribution to this work made by the late Dr. Atsushi Matsui.

References

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations Session at EACL 2012*, Avignon, France. Association for Computational Linguistics. Available from `http://brat.nlplab.org`.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Peter Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Wei Xu, Chris Callison-Burch, and Bill Dolan. 2015. SemEval-2015 task 1: Paraphrase and semantic similarity in twitter (PIT). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 1–11, Denver, Colorado. Association for Computational Linguistics.