Convolutional Neural Networks for Churn Prediction in Microblogs

Mourad Gridach
Department of Computer Science
High Institute of Technology
Ibn Zohr University, Agadir
Morocco

Abstract

For brands, gaining new customer is more expensive than keeping an existing one. Therefore, the ability to keep customers in a brand is becoming more challenging these days. Churn happens when a customer leaves a brand to another competitor. Most of the previous work considers the problem of churn prediction using the Call Detail Records (CDRs). In this paper, we use micro-posts to classify customers into churny or non-churny. In recent years, Deep Neural Networks (DNNs) achieved state-of-the-art in various NLP applications. We investigate the use of convolutional neural networks (CNNs) to classify customers to churny or non-churny. In addition, we show that using pretrained word embeddings to initialize word vectors improve the system performance. Experimental results showed that we were able to outperform the state-of-the-art results on publicly available Twitter dataset with an interesting margin.

1 Introduction

Customer churn may be defined as the process of losing a customer that just recently switches from a brand to another competitor. The churn problem can be tackle from different angles: most of the previous work use CDRs to identify churners from non-churners (add references). More recently, with more data became available on the web, brands can use opinions expressed by customers on social networks, forums and especially Twitter to discriminate churny from non-churny customers. We used the churn dataset developed by (Amiri and Daumé III, 2015) and collected from Twitter for three telecommunication brands: Verizon, T-Mobile, and AT&T.

Previous state-of-the-art works tackled the churn using machine learning techniques (linear classification, support vector machines, and logistic regression) with hand-crafted features(Amiri and Daumé III, 2015). More recently,(Amiri and Daumé III, 2016) used Recurrent Neural Networks (RNNs) to classify customers in churners and non-churners.

Recently, deep learning models have achieved great success in various domains and difficult problems such as computer vision (Krizhevsky et al., 2012) and speech recognition (Hinton et al., 2012). In natural language processing, much of the work with deep learning models has involved language modeling (Bengio et al., 2003; Mikolov et al., 2013), sentiment analysis (Socher et al., 2013; Dos Santos and Gatti, 2014), and more recently, neural machine translation (Cho et al., 2014; Sutskever et al., 2014). Furthermore, these models can use backpropagation algorithm for training (Rumelhart et al., 1988).

We investigate the use of convolutional neural networks (CNN) combined with pretrained word embeddings to predict the churny from non-churny customers in micro-blogs. CNN achieved astonishing results in various applications in computer vision (add references). In addition, CNN has been shown to be effective in many NLP applications, achieving better results in sentence modeling(Kalchbrenner et al., 2014), search query retrieval (Shen et al., 2014) and neural language models (Kim et al., 2015).

2 Approach

Our model is based on a CNN where we initialize word vectors with pretrained word embeddings which has been used in various NLP applications such as NER (Ma and Hovy, 2016). The pretrained word embeddings are publicly available, they were trained by (Mikolov et al., 2013). We show that the same idea can be applied to churn prediction in micro-blogs to improve the system performance. This is consistent with the idea that in many NLP classification tasks, pretrained word embeddings are universal feature extractors.

The main architecture used in this paper is a variant of the model used by (Collobert et al., 2011). Given a tweet T with length n where we add padding whenever it is necessary for the model, T is represented as the following:

$$v_n^1 = v_1 \oplus v_2 \oplus ... \oplus v_n \quad (1)$$

where v_i represents the word vector of the i-th word in the sentence T and $\oplus$ represents the concatenation operator. We use successive filters w to obtain multiples feature map. Each filter is applied to a window of m words to get a single feature map: $F_i = f(w.v_{i+m-1}^i + b)$ where b is the bias and f is the non-linearity where we used ReLU (Rectified Linear Unit). In the next step, we applied a max-over-time pooling operation (Collobert et al., 2011) to the feature map and take the maximum value. The results are feed to a fully connected softmax layer to get probabilities over the tweets. Figure 1 illustrates the architecture of our system where we consider the system is classifying the input sentence: “swtich from crappy Brand-1 to Brand-2 or Brand-3”.

2.1 Implementation details

Training is done using stochastic gradient descent over mini-batches with the Adadelta update rule (Zeiler, 2012). The windows used are $m = 3, 4$ and 5 with 100 feature maps each, the mini-batch size is 50 and dropout rate is 0.5. We used 10% of the training dataset as dev set to tune the hyper-parameters. For pretrained word vectors, we use the publicly available word2vec vectors trained on 100 billion words from Google News. We used 300 as the dimension of word vectors using the continuous bag-of-words model.

3 Dataset and Experiments

We use the dataset provided by (Amiri and Daumé III, 2015). The authors collected the data from twitter for three telecom brands: Verizon, T-Mobile, and AT&T (Table 1). We test our model using this dataset and compare the obtained results with the previous best results. The state-of-the-art results were produced by (Amiri and Daumé III, 2016) where they achieved 78.30 in F1 score. They used a combination of Bag of Words (BOW) features and Recurrent Neural Networks (RNN). Table 2 shows a brief presentation of the experimental results and the comparison with the previous best system. We were able to achieve the state-of-the-art by outperforming the previous best system by 2.37 in F1-score.

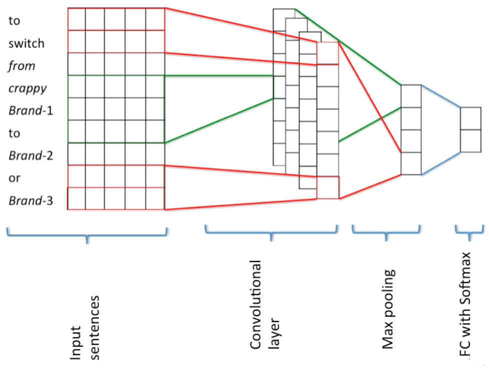

Figure 1: The system architecture.

Brand	Churny	Non-Churny
Verizon	447	1543
T-Mobile	95	978
AT&T	402	1389

Table 1: The churn microblog dataset.

Models	F1-score
Amiri and Daume III, 2016	78.30
ChurnCNN	80.67

Table 2: Comparison between our system (ChurnCNN) and (Amiri and Daumé III, 2015).

In the work of (Amiri and Daumé III, 2016), authors showed that state-of-the-art approaches used in sentiment analysis fail to identify churny content. In this paper, we showed that an end-to-end deep CNN with pretrained word embeddings outperforms all the previous approaches. It should be noted that we used a simple CNN without hand-engineered features. In the future work, we will use more complex deep neural network architecture in order to improve our results.

References

Hadi Amiri and Hal Daumé III. 2015. Target-dependent churn classification in microblogs. In *AAAI*. pages 2361–2367.

Hadi Amiri and Hal Daumé III. 2016. Short text representation for detecting churn in microblogs. In *AAAI*. pages 2566–2572.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* .

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12(Aug):2493–2537.

Cícero Nogueira Dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*. pages 69–78.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29(6):82–97.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* .

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2015. Character-aware neural language models. *arXiv preprint arXiv:1508.06615* .

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. pages 1097–1105.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. Learning representations by back-propagating errors. *Cognitive modeling* 5(3):1.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, pages 373–374.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, Christopher Potts, et al. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*. Citeseer, volume 1631, page 1642.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .