Churn Identification in Microblogs using Convolutional Neural Networks with Structured Logical Knowledge

Mourad Gridach
Department of Computer Science
High Institute of Technology
Ibn Zohr University, Agadir
Morocco
m.gridach@uiz.ac.ma

Hatem Haddad
Department of Computer and Decision Engineering, Université Libre de Bruxelles
Belgium
Hatem.Haddad@ulb.ac.be

Hala Mulki
Department of Computer Engineering
Selcuk University, Konya
Turkey
hallamulki@gmail.com

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 CDRs. In this paper, we use micro-posts to classify customers into churny or non-churny. We explore the power of CNNs since they achieved state-of-the-art in various computer vision and NLP applications. However, the robustness of end-to-end models has some limitations such as the availability of a large amount of labeled data and uninterpretability of these models. We investigate the use of CNNs augmented with structured logic rules to overcome or reduce this issue. We developed our system called $Churn\_teacher$ by using an iterative distillation method that transfers the knowledge, extracted using just the combination of three logic rules, directly into the weight of Deep Neural Networks (DNNs). Furthermore, we used weight normalization to speed up training our convolutional neural networks. Experimental results showed that with just these three rules, we were able to get state-of-the-art on publicly available Twitter dataset about three Telecom brands.

1 Introduction

Customer churn may be defined as the process of losing a customer that recently switches from a brand to another competitor. The churn problem can be tackled from different angles: most of the previous work used Call Detail Records (CDRs) to identify churners from non-churners (Zaratiegui et al., 2015). More recently, with more data became available on the web, brands can use customers opinionated comments via social networks, forums and especially Twitter to detect churny from non-churny customers. We used the churn dataset developed by (Amiri and Daumé III, 2015). This dataset was collected from Twitter for three telecommunication brands: Verizon, T-Mobile, and AT&T.

In recent years, deep learning models have achieved great success in various domains and difficult problems such as computer vision (Krizhevsky et al., 2012) and speech recognition (Dahl et al., 2012; 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), 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).

Regardless of the success of deep neural networks, these models still have a gap compared to human learning process. While the success came from the high expressiveness, it leads them to predict results in uninterpretable ways, which could have a negative side effects on the whole learning process (Szegedy et al., 2013; Nguyen et al., 2015). In addition, these models require a huge amount of labeled data, which is considered as time consuming for the community since it requires human experts in most applications (natural language and computer vision). Recent works tackled this issue by trying to bridge this gap in different applications: in supervised learning such as machine translation (Wu et al., 2016) and unsupervised learning (Bengio et al., 2015).

In the Natural Language Processing (NLP) community, there has been much work to augment the training process with additional and useful features (Collobert et al., 2011) which proved its success in various NLP applications such as Named Entity Recognition (NER) and Sentiment Analysis. The majority of these works used pretrained word embeddings obtained from unlabeled data to initialize their word vectors, which allow them to improve the performance. Another solution came from integrating logical rules extracted from the data directly into the weights of neural networks. (Hu et al., 2016) explored a distillation framework that transfers structured knowledge coded as logic rules into the weights of neural networks. (Garcez et al., 2012) developed a neural network from a given rule to do reasoning.

In this paper, we combine the two ideas: firstly, we used unsupervised word representations to initialize our word vectors. We explored three different pretraiend word embeddings and compared the results with a randomly sampled one. Secondly, we used three main logic rules, which were proven to be useful and crucial. The "but" rule was explored by (Hu et al., 2016) in sentiment analysis and we add two new rules: "switch to" and "switch from". (Amiri and Daumé III, 2016) showed that the last two rules have a remarkable influence into the churn classification problem.

Moreover, in order to accelerate training our model on churn training dataset, we conduct an investigation of using weight normalization (Salimans and Kingma, 2016), which is a new recently developed method to accelerate training deep neural networks.

Experiments on Twitter dataset built from a large number of tweets about three Telecommunication brands show that we were able to obtain state-of-the-art results for churn classification in microblogs. Our system, called $Churn\_teacher$, is constructed by using a structured logical knowledge expressed into three logic rules transferred into the weights of convolutional neural networks. We outperform the previous models based on hand engineering features or also using recurrent neural networks combined with minimal features. Our system is philosophical close to (Hu et al., 2016), which showed that combining deep neural networks with logic rules performed well on two NLP tasks: NER and Sentiment Analysis.

The rest of this paper is structured as follows: in section 2, we discuss the related work done in churn prediction application. In section 3, we present our churn prediction approach which is based on structured logical knowledge transferred into the weights of Convolutional Neural Networks (CNNs). In section 4, we discuss the impact of pretrained word embeddings on the churn classification. The experimental results are presented in section 5. Finally, we present the conclusion with the future work in section 6.

2 Related Work

Churn prediction is an important area of focus for sentiment analysis and opinion mining. In the 2009, ACM Conference on Knowledge Discovery and Datamining (KDD) hosted a competition on predicting mobile network churn using a large dataset posted by Orange Labs, which makes churn prediction, a promising application in the next few years. We can divide the previous work on Customer churn prediction in two research groups: the first group uses data from companies such as Telecom providers, banks, or other organizations. More recently, with the explosion of social networks, researchers are interested to use social networks such as Twitter to predict churners.

Using data from banks, (Keramati et al., 2016) developed a system for a customer churn in electronic banking services. They used a decision tree algorithm to build their classification model. The main goal of this paper is studying the most relevant features of churners in banking services such as demographic variables (age, gender, career,

etc.), transaction data through electronic banking portals (ATM, mobile bank, telephone bank, etc.), and others. They used a method called CRISP-DM which contains six phases: Business understanding, Data understanding, Data preprocessing, Modeling, Evaluation and Deployment. At the final stage, they used a decision tree method to model the previous phases.

(Backiel et al., 2016) studied the impact of incorporating social network information into churn prediction models. The authors used three different machine learning (ML) techniques: logistic regression, neural networks and Cox proportional hazards. To extract features to use with these ML techniques, they built a call graph, which allowed them to extract the relevant features.

(Li et al., 2016) developed a model based on stacked auto-encoder as a feature extractor to detect the most influential features in Telecom churn prediction. In addition, they proposed a second model where they augmented the previous model with a Fishers ration analysis called Hybrid Stack Auto-Encoder (HSAE). The models were evaluated on Orange datasets. Experimental results showed that the HSAE model outperformed all the other models including Principal Component Analysis (PCA).

More recently, researchers tackle the churn prediction problem using data collected from microblogs. (Amiri and Daumé III, 2015) developed a system for churn prediction in microblogs. They investigated the machine learning models such as support vector machines, and logistic regression with the combination of extracted features. Furthermore, they investigated the use of three different churn indicators: demographic, content, and context indicators. Experimental results showed that the combination of the three indicators lead to the best performance.

(Amiri and Daumé III, 2016) used the power of Recurrent Neural Networks (RNN) as a representation learning models in order to learn micropost and churn indicator representations. The experiments on publicly available Twitter dataset showed the efficiency of the proposed method in classifying customers in churners and non-churners. Moreover, authors showed that the churn classification problem is different from classical sentiment analysis problem since the previous state-of-the-art sentiment analysis systems failed to classify churny/non-churny customers.

In this work, we focus on churn prediction in microblogs where we use a publicly available Twitter dataset provided by (Amiri and Daumé III, 2015) to evaluate our system.

3 The Proposed System

In this section, we introduce our system that enables a convolutional neural network to learn simultaneously from logic rules and labeled data in order to classify customers as churners and non-churners. The general architecture of our system can be seen as the combination of the knowledge distillation (Hinton et al., 2015) and the posterior regularization method (Ganchev et al., 2010). (Hu et al., 2016) explored this combination to build two systems for English sentiment analysis and named entity recognition. We show that this framework can also be applied to customer churn prediction by deriving more logic rules and transfer the structured logical knowledge into the weights of a convolutional neural network.

3.1 Problem Formulation

For the purpose of this paper, let us assume we have $x \in \mathcal{X}$ is the input variable and $y \in \mathcal{Y}$ is the target variable. Let us consider the training data $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$ which is a set of instantiations of (x, y) and N is the number of training examples in our dataset. For the purpose and the clarity of this paper, we focus on classification problem where the target y is a one-hot encoding of the class labels. We consider a subset of the training data $(X, Y) \subset (\mathcal{X}, \mathcal{Y})$ as a set of instances of (x, y). A neural network defines a conditional probability $p_\theta(y|x)$ parameterized by θ.

3.2 Neural Network with Structured Logical Knowledge

In this section, we describe the distillation method that allowed our system to transfer the structured logical knowledge into the weights of convolutional neural networks to classify customers as churners and non-churners.

Let us define the set of constraint functions f_l such that: $f_l \in \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$, where l is the index of a specific constraint function. In our problem, the set of functions will be represented as logical rules where the overall truth values are in the interval $[0, 1]$. These functions will allow us to encode the structured logical knowledge where the goal is

to satisfy (i.e., maximize by optimizing the predictions y) with confidence weights $\lambda_l \in \mathcal{R}$.

The construction of a structure-enriched teacher network q at each iteration from the neural network parameters is obtained by solving the following optimization problem:

$$\min_{q \in \mathcal{P}} KL(q(Y)||p_\theta(Y|X)) - C \sum_l \lambda_l \, \mathbb{E}_q[f_l(X,Y)] \quad (1)$$

where $\mathcal{P}$ denotes the appropriate distribution space; and C is the regularization parameter. In this paper, the teacher is called $Churn\_teacher$ where its main goal is to teach the model to classify customers from churners and non-churners. The closeness between our $Churn\_teacher$ and p_θ, which represents the conditional probability obtained by the softmax output layer of the convolutional neural network, is measured using the Kullback-Leibler (KL) divergence where the aim is minimizing it. We note that problem (1) is convex and has a closed-form solution given by the following:

$$q^*(Y) \propto p_\theta(Y|X) \exp\{C \sum_l \lambda_l f_l(X,Y)\} \quad (2)$$

It should be noted that the normalization term can be computed efficiently through direct enumeration of the chosen rule constraints. At each iteration, the probability distribution of the neural network p_θ is updated using the distillation objective (Hinton et al., 2015) that balances between imitating soft predictions of our $Churn\_teacher$ q and predicting true hard labels:

$$\theta^{(t+1)} = arg \min_{\theta \in \Theta} \frac{1}{N} \sum\nolimits_{n=1}^{N} (1-\pi) l(y_n, \sigma_\theta(x_n)) + \pi l(sp_n^{(t)}, \sigma_\theta(x_n)) \quad (3)$$

where l denotes the cross entropy loss function that we used in this paper; N is the training size; $\sigma_\theta(x)$ is the softmax output of p_θ on x; sp_n is the soft prediction vector of the $Churn\_teacher$ q on training point x_n at iteration t and π is the imitation parameter calibrating the relative importance of the two objectives. In addition to their teacher q, (Hu et al., 2016) tested their results using a student network p. Related to their experimental results, the teacher q always gives better results than the student p. We decide to only use the teacher network q called in our work the $Churn\_teacher$.

3.3 Neural Network Architecture

In this section, we give an overview of the convolutional neural network used in our work. The main architecture of our CNN is depicted in Figure 1. We followed the convolutional neural network architecture as proposed by (Kim, 2014).

In the first step, we initialize each word in a sentence T with length n with pretrained word representations learned from unannotated corpora. Then, we add padding whenever it is necessary for the model. T is represented as the following:

$$v_1^n = v_1 \oplus v_2 \oplus ... \oplus v_n \quad (4)$$

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: $F_i = \phi(w.v_i^{i+m-1} + b)$ where b is the bias and ϕ denotes an elementwise nonlinearity 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 fed to a fully connected softmax layer to get probabilities over the sentences. Figure 1 illustrates the architecture of our system where we consider the system is classifying the input sentence: "Damn thats crud. You should switch to Verizon".

3.4 Logic Rules

In the early stages of the expansion of artificial intelligence, (Minsky, 1983) argued that the cognitive process of human beings learn from two different sources: examples as deep neural networks are doing these days and also from rich experiences and general knowledge. For this reason, we will use both of the sources for churn prediction in microblogs where the convolutional neural network learns from examples and logic rules add structured knowledge into the weights of CNN by playing a role of a regularizer in the learning process.

In this section, we present the three logic rules that we used in our churn prediction system. We

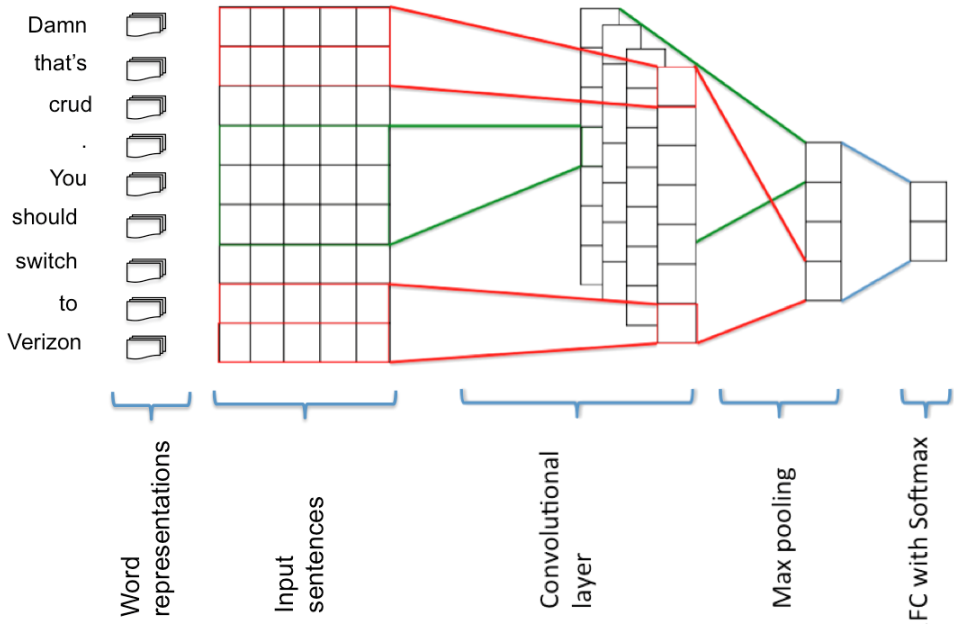

Figure 1: The system architecture. The word vectors are initialized with pretrained word representations from one of the three models: GloVe, Skip-Gram or CBOW. We feed these word vectors to the convolutional neural network as input sentences followed by a max-pooling overtime followed by a fully-connected layer and softmax to get probabilities (FC with Softmax)

borrow the first logic rule from sentiment analysis literature by using the conjunction word "but". It has been shown that "but" played a vital role in determining the sentiment of a sentence where the overall sentiment is dominated by the clauses following the word "but" (Jia et al., 2009; Dadvar et al., 2011; Hogenboom et al., 2011; Hu et al., 2016). For a given sentence with "$C1$ but $C2$", we assume that the sentiment of the whole sentiment will take the polarity of clause $C2$.

The second logic rule that we used is "switch from" considered as a target-dependent churn classification rule. (Amiri and Daumé III, 2016) showed that "switch from" can have an important role to classify if the customer will be churner or non-churner. The last logic rule that we explored in our work is similar to the second rule for being target dependent churn classification where we substitute the preposition "from" to be the preposition "to" to obtain "switch to". For a given sentence with the form "$C1$ switch to $C2$", it is clear that the customer will choose to switch to the brand present in clause $C2$.

Consider the two examples from the training data:

- Damn thats crud. You should switch to Verizon.
- Gonna switch from bad @verizon internet to @comcast. @VerizonFiOS will never be in my area and i bet @googlefiber will get here first.

In the first example, the customer will switch to the brand "Verizon" while for the second example, the customer will leave the brand "Verizon" to another competitor. Consequently, with respect to the brand "Verizon", the first tweet is classified as "Non-churny" and the second tweet is classified as "Churny".

To encode these rules, we used soft logic (Bach et al., 2015) where we can represent truth values from the interval [0, 1]. The main Boolean operators are formulated as the following (Foulds et al., 2015):

$$
\begin{aligned}
&X \& Y = max\{X + Y - 1, 0\} \\
&X \vee Y = min\{X + Y, 1\} \\
&X_1 \wedge ... \wedge X_N = \sum_i \frac{X_i}{N} \\
&\neg X = 1 - X
\end{aligned}
\tag{5}
$$

This logic rule "but" is written as follows:

$$has \text{ "C1-but-C2" } structure(S) \implies [1(y = +) \implies \sigma_\theta(C2)_+ \wedge \sigma_\theta(C2)_+ \implies 1(y = +)] \quad (6)$$

In the equation above, $1(.)$ represents an indicator function that can take two values: 1 if the argument is true, and 0 otherwise. Class "+" represents "positive"; and $\sigma_\theta(C2)_+$ is the element of $\sigma_\theta(C2)$ for class "+". Following the Equation 5, when S has the "C1- but-C2" structure, the truth value of the above logic rule equals to $(1 + \sigma_\theta(C2)_+)/2$ when y = "+", and $(2 - \sigma_\theta(C2)_+)/2$ otherwise.

For the two other logic rules "switch to" and "switch from", we followed the same structure of the "but" rule with a slightly different settings:

$$has \text{ "C1-switch to/from-C2" } structure(S) \implies [1(y = +) \implies \sigma_\theta(C2)_+ \wedge \sigma_\theta(C2)_+ \implies 1(y = +)] \quad (7)$$

For the logic rule "switch to", if we are classifying the sentence S with respect to a brand in the clause $C2$, then the argument will be true which gives the formula: $(1 + (\sigma_\theta(C2)_+)/2$. The logic rule "switch from" plays an opposite role where if a brand is in clause $C2$, the overall sentiment will be negative with respect to this brand, so we use this formula: $(2 - \sigma_\theta(C2)_+)/2$.

3.5 Training Details

Training is done using stochastic gradient descent over mini-batches with the Adadelta update rule (Zeiler, 2012). Word vectors are initialized using pretrained word embeddings and fine-tuned throughout training. At each iteration of the training process, we enumerate the rule constraints (or a set of rules if we use them all at once) in order to compute the soft predictions of our $Churn\_teacher$ q by using the equation 2. During the experiments, we choose the imitation parameter to be $\pi(t) = 1 - 0.85t$ and the regularization parameter to be $C = 100$. We set the confidence levels of rules to be $\lambda_l = 1$. It should be noted that we used the model results on development set in order to select the best hyperparameters. The training procedure is summarized in algorithm 1.

Algorithm 1: Churn prediction using CNN with logic rules

Input: The training data $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$

The rule set $R = \{(R_l, \lambda_l)\}_{l=1}^{3}$

Parameters: π - imitation parameter

C regularization strength

Initialize neural network parameters

Choose the rule Rl or a set of rules

do

Sample a minibatch (X, Y) $\subset \mathcal{D}$

Build the $Churn\_teacher$ network using Equation 2

Update p_θ using Equation 3

while not converged

Output: Churn_teacher q

3.6 The Effect of Weight Normalization

It should be noted that we use weight normalization which is a new method introduced by (Salimans and Kingma, 2016) to reparameterize the weight vectors in a deep neural network in order to decouple the length of those weight vectors from their direction. Using this method, the authors showed that they were able to improve the conditioning of the optimization problem and speed up convergence of stochastic gradient descent. This method followed earlier work by (Ioffe and Szegedy, 2015) where they introduced batch normalization trying to normalize each neuron output by the mean and standard deviation of the outputs computed over a minibatch of examples. More recently, this method was widely used in deep learning architectures such as deep convolutional neural networks, deep reinforcement learning, generative adversarial networks (GANs) and others (Smith and Topin, 2016; Gehring et al., 2017). Using weight normalization in training our convolutional neural network allowed us to accelerate convergence of stochastic gradient descent.

4 Input Word Embeddings

The research in representations of words as continuous vectors has a long history where many ideas were proposed (Hinton et al., 1985). More recently, (Bengio et al., 2003) proposed a model architecture based on feedforward neural networks for estimating neural network language model. The most popular model for word representations was developed by (Mikolov et al., 2013) called word2vec where they used either of two model architectures to produce a distributed representation of words: continuous bag-of-words (CBOW) model or Skip-Gram (SG) model. Another popular model for word representations developed by (Pennington et al., 2014) called "GloVe" (Global Vectors). The main difference between

Base NN	Word Embeddings Model	F1 score
CNN	Random embeddings	77.13
CNN	Continuous bag-of-words	79.89
CNN	Skip-Gram	79.55
CNN	GloVe	80.67

Table 1: Results with different choices of pretrained word embeddings with a comparison with randomly initialized ones on churn predicition in microblogs

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

Table 2: Statistics and details about the churn microblog dataset.

this model and word2vec models is the representations of a word in vector space: word2vec models use a window approach while GloVe uses the global statistics of word-word co-occurrence in the corpus to be captured by the model.

(Hisamoto et al., 2013) used word embeddings features for English dependency parsing where they employed flat (non-hierarchical) cluster IDs and binary strings obtained via sign quantization of the vectors. For chunking, (Turian et al., 2010) showed that adding word embeddings allows the English chunker to increase its F1-score. (Huang et al., 2014) showed that adding word embeddings as features for English part-of-speech (POS) tagging task helped the model to increase its performance. (Bansal et al., 2014) argued that using word embeddings in parsing English text improved the system performance. For English sentiment analysis, (Kim, 2014) showed that using pretrained word embeddings helped their system to improve its accuracy.

As in (Collobert et al., 2011), in order to test the importance of pretrained word embeddings in churn prediction for microblogs, we performed experiments with different sets of publicly published word embeddings, as well as a random sampling method, to initialize word vectors in our model. We investigate three different pretrained word embeddings: Skip-Gram, continuous bag-of-words and Stanford's GloVe model. Table 1 gives the performance of three different word embeddings, as well as the randomly sampled one. It should be noted that the random embeddings are uniformly sampled from range $[-\sqrt{\frac{3}{dim}}, +\sqrt{\frac{3}{dim}}]$, where dim is the dimension of embeddings. According

Models	Precision	Recall	F1
CNN	75.36	79	77.13
CNN + pretrained	79.28	82.11	80.67
CNN-pre + but	80.84	83.09	81.95
CNN-pre + switch from	79.74	82.14	80.92
CNN-pre + switch to	80.89	84.39	82.60
CNN-pre + all the 3 rules	82.56	85.18	83.85

Table 3: Results with the three logic rules compared to the without and with word embeddings. We test the model by using each rule independently and after that we combine them in one experiment.

to the results in Table 1, we see that using pretrained word embeddings obtain a significant improvement, about 3.54% in F1 score as opposed to the ones using random embeddings. This is consistent with results reported by previous work done in other NLP tasks (Collobert et al., 2011; Huang et al., 2015; Chiu and Nichols, 2015).

For different pretrained embeddings, Stanford's GloVe 300 dimensional embeddings achieves best results, about 1.12% better than Skip-gram model and 0.78% better than continuous bag-of-words model. One possible reason that Word2Vec is not as good as the Stanford's GloVe model is because of vocabulary mismatch, since Word2Vec embeddings were trained in case-sensitive manner, excluding many common symbols such as punctuations and digits. Because we do not use any kind of data pre-processing to deal with such common symbols or rare words, it might be an issue for using Word2Vec.

5 Experimental Results

For the evaluation of our model, 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 2 presents the details about the entries of this dataset. We divide the experimental process into two stages: the first stage concerns running the experiments using the convolutional neural network without and with different logic rules in order to select the best achieved results. In the second stage, we compare our best settings with the previous state-of-art system in churn prediction in microblogs.

Table 3 shows the churn classification results. The first row represents the baseline where we use only the convolutional neural network. In the second row, we initialize our word vectors using

Rule	Number of rules in the training set	Number of rules in the test set
But	358	83
Switch to	321	71
Switch from	62	11
Total	741	165

Table 4: Statistics about the three rules (but, switch to and switch from) in the training and test set.

pretrained word vectors using GloVe model since it gives us the best results among the other pretrained word vectors. We get an improvement around 2.5% in F1-score. We refer to this model by "CNN-pre". This results is consistent with the fact that these pretrained word vectors are universal feature extractors that shown an important results in different NLP applications such as sentiment analysis, named entity recognition and Part-of-speech tagging.

By transferring the knowledge extracted using the "but" logic rule into the weights of the convolutional neural network, we were able to improve the F1-score over the CNN-pre model by 1.28 points in F1-score. For the "switch from" logic rule, we get a slight improvement over the CNN-pre model by 0.25 points in F1-score. The biggest improvement among the three logic rules was obtained by the "switch to" rule where we were able to improve the performance over the CNN-pre model by 1.93 points in F1-score. The last row in Table 3 concerns the results that we obtained by using all the three logic rules where logically we achieved best results and outperformed the CNN-pre model by 3.18 points in F1-score.

While we do not have a complete explanation why we got better results with "switch to" rule, we believe that it is caused by the fact that there 321 sentences in the training data containing this rule which represents around 8% sentences contains the word "switch to". Moreover, it will be clear that customer will leave a specific brand to another new brand. For the "switch from" rule, we get slight improvement over the CNN-pre model because few sentences containing this rule (around 2% sentences contains the word "switch from").

Table 4 shows the statistics about the presence of the three rules in the training data. For "but" rule, we also get an important improvement over the CNN-pre model which confirms the results obtained by (Hu et al., 2016) using the same rule in sentiment analysis. We note that around 9% sentences contains the word "but". In the last row, we combine all the three rules and we were able to obtain the best performance. We refer to this model as "$Churn\_teacher$". This is consistent with the argument by (Hu et al., 2016) where they argued that more rules will allow the system to improve its performance over the base convolutional neural network.

Models	F1 score
Unigram + Nb (Amiri and Daumé III, 2015)	73.42
(Amiri and Daumé III, 2016)	78.30
Churn_teacher	83.85

Table 5: Comparison of our system with two previous systems.

We test our model using this dataset and compare the obtained results with two other systems. 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 features and Recurrent Neural Networks. The second system referenced here as "Unigram + Nb" developed by (Amiri and Daumé III, 2015) used different N-grams ($n = 1, 2, 3$) and their combination on both of the word and character levels.

By adding three rules to the convolutional neural networks, we outperformed the "Unigram + Nb" system by a large margin (10.43 points in F1-score). Furthermore, our model also outperformed the system developed by (Amiri and Daumé III, 2016) by a good margin (5.55 points in F1-score). Table 5 shows a brief presentation of the experimental results and the comparison with the two other systems.

6 Conclusion

In this paper, we explored the problem of target-dependent churn classification in microblogs. We combine the power of convolutional neural networks with structured logic knowledge by constructing a churn teacher capable of classifying customers into churners and non-churners. In addition, we confirm that initializing word vectors with pretrained word embeddings trained on unannotated corpora improved the system performance.

A key aspect of our system is that it explores the transfer of the structured knowledge of logic rules into the weights of convolutional neural networks for churn classification problem. By com-

bining three logic rules, our model largely outperformed all the previous models on publicly available Twitter dataset. We showed that "but" rule is also useful for churn prediction to confirm the results obtained for sentiment classification problem. We consider the two other rules ("switch to" and "switch from") as target-specific rules for churn classification problem which helped the system to improve its performance.

In the future work, we will explore the use of character-level embeddings where we will represent word in a sentence by a word vector representing the concatenation of two embeddings: its equivalent word embeddings obtained from the lookup table and the embeddings obtained from its characters. Furthermore, we will explore the use of named entity recognition to recognize different organizations where we will focus on brands which we believe could help us to a better churn classification.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. We would like to thank Doctor Hadi Amiri and Doctor Hal Daume III for providing us with Twitter dataset for churn prediction in microblogs.

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.

Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. 2015. Hinge-loss markov random fields and probabilistic soft logic. *arXiv preprint arXiv:1505.04406* .

Aimée Backiel, Bart Baesens, and Gerda Claeskens. 2016. Predicting time-to-churn of prepaid mobile telephone customers using social network analysis. *Journal of the Operational Research Society* 67(9):0.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *ACL (2)*. pages 809–815.

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

Yoshua Bengio, Dong-Hyun Lee, Jorg Bornschein, Thomas Mesnard, and Zhouhan Lin. 2015. Towards biologically plausible deep learning. *arXiv preprint arXiv:1502.04156* .

Jason PC Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308* .

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.

Maral Dadvar, Claudia Hauff, and FMG De Jong. 2011. Scope of negation detection in sentiment analysis .

George E Dahl, Dong Yu, Li Deng, and Alex Acero. 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing* 20(1):30–42.

James Foulds, Shachi Kumar, and Lise Getoor. 2015. Latent topic networks: A versatile probabilistic programming framework for topic models. In *International Conference on Machine Learning*. pages 777–786.

Kuzman Ganchev, Jennifer Gillenwater, Ben Taskar, et al. 2010. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research* 11(Jul):2001–2049.

Artur S d'Avila Garcez, Krysia Broda, and Dov M Gabbay. 2012. *Neural-symbolic learning systems: foundations and applications*. Springer Science & Business Media.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122* .

GE Hinton, DE Rumelhart, and RJ Williams. 1985. Learning internal representations by back-propagating errors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* 1.

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.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

Sorami Hisamoto, Kevin Duh, and Yuji Matsumoto. 2013. An empirical investigation of word representations for parsing the web. In *Proceedings of ANLP*. pages 188–193.

Alexander Hogenboom, Paul Van Iterson, Bas Heerschop, Flavius Frasincar, and Uzay Kaymak. 2011. Determining negation scope and strength in sentiment analysis. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*. IEEE, pages 2589–2594.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. *arXiv preprint arXiv:1603.06318* .

Fei Huang, Arun Ahuja, Doug Downey, Yi Yang, Yuhong Guo, and Alexander Yates. 2014. Learning representations for weakly supervised natural language processing tasks. *Computational Linguistics* 40(1):85–120.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991* .

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* .

Lifeng Jia, Clement Yu, and Weiyi Meng. 2009. The effect of negation on sentiment analysis and retrieval effectiveness. In *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, pages 1827–1830.

Abbas Keramati, Hajar Ghaneei, and Seyed Mohammad Mirmohammadi. 2016. Developing a prediction model for customer churn from electronic banking services using data mining. *Financial Innovation* 2(1):10.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* .

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.

Ruiqi Li, Peng Wang, and Zonghai Chen. 2016. A feature extraction method based on stacked autoencoder for telecom churn prediction. In *Asian Simulation Conference*. Springer, pages 568–576.

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.

Marvin Minsky. 1983. *Learning meaning*. Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Anh Nguyen, Jason Yosinski, and Jeff Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 427–436.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–1543.

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

Tim Salimans and Diederik P Kingma. 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*. pages 901–901.

Leslie N Smith and Nicholay Topin. 2016. Deep convolutional neural network design patterns. *arXiv preprint arXiv:1611.00847* .

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)*. 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.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* .

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, pages 384–394.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* .

Jaime Zaratiegui, Ana Montoro, and Federico Castanedo. 2015. Performing highly accurate predictions through convolutional networks for actual telecommunication challenges. *arXiv preprint arXiv:1511.04906* .

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