A comparison between CNNs and WFAs for sequence classification

Ariadna Quattoni
Universitat Politècnica de Catalunya
Campus Nord, Barcelona
aquattoni@cs.upc.edu

Xavier Carreras
IIIA-CSIC
Campus UAB, Bellaterra
xavierc@iiia.csic.es

Abstract

We compare a classical CNN architecture for sequence classification involving several convolutional and max-pooling layers against a simple model based on weighted finite state automata (WFA). Each model has its advantages and disadvantages and it is possible that they could be combined. However, we believe that the first research goal should be to investigate and understand how do these two apparently dissimilar models compare in the context of specific natural language processing tasks. This paper is the first step towards that goal. Our experiments with five sequence classification datasets suggest that, despite the apparent simplicity of WFA models and training algorithms, the performance of WFAs is comparable to that of the CNNs.

1 Introduction

In the latter years CNNs have been proposed as models for sequence classification and it has been shown that they can give competitive results, even when compared to more complex models (Kim, 2014; Zhang and Wallace, 2017; Kalchbrenner et al., 2014; Johnson and Zhang, 2015; Goldberg, 2016). They typically combine various convolutional filters with max-pooling layers.

Because they have several interacting layers, it is in general is hard to interpret exactly what is it that they are learning. But most likely their success relies on the fact that their convolutional filters have the ability to capture arbitrary features of the input sequence.

On the other hand, non-deterministic weighted automata (WFAs) are recurrent models that only use linear activation functions.Essentially, WFAs can be regarded as recurrent neural networks where the function that predicts the dynamic state representation from previous states is linear.

For more details about the relations between linear activation RNNs and WFAs, we refer the reader to (Rabusseau et al., 2019). Several algorithms based on low rank matrix decompositions have been proposed (Hsu et al., 2009, 2012; Bailly et al., 2009; Balle et al., 2011; Cohen et al., 2012; Balle et al., 2014).

In addition to being easily trainable, WFAs offer other advantages. The main advantage is that they are classical computer science models that have been intensively researched in the theoretical community. Because of this they are relatively well understood and we know how to efficiently perform important computations. For example, consider a WFA computing a distribution over strings, there are simple and efficient algorithms to compute marginal probabilities for prefixes, infixes and suffixes. Furthermore another advantage of these models is that there are well known and understood algorithms for transforming them into deterministic automata. The resulting deterministic automata can be used to interpret the computation performed by WFAs.

Both CNNs and WFAs are general models, and the exact architecture can be specified to solve different tasks such as language modeling, or sequence classification which is the focus of this paper. Each model has its advantages and disadvantages and it is possible that they could be combined.

However, we believe that the first research goal should be to investigate and understand how do these two apparently dissimilar models compare in the context of specific natural language processing tasks.

This paper is the first step towards that goal. We focus on the task of sequence classification and compare the performance of WFAs and CNNs trained under the same initial conditions, over five different data sets.

To a certain extent a similar comparison between

Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing, pages 159–163
Online, November 20, 2020.

WFAs and neural models was made by (Quattoni and Carreras, 2019) in the context of language modeling. But to our knowledge this is the first empirical comparison of CNNs and WFAs for sequence classification.

2 WFAs for Sequence Classification

2.1 Preliminaries: WFAs for sequence modelling

We will use weighted finite state automata (WFAs) as elementary building blocks to build our sequence prediction model.

More precisely, a WFA takes as input a sequence and outputs a real number, that is: $f : \Sigma^\star \rightarrow \mathbb{R}$ where $x = x_1 \cdots x_n$ is sequence of length n over some finite alphabet Σ.

We denote as $\Sigma^\star$ the set of all finite sequences, and we use it as a domain of our functions. A WFA with k states is defined as a tuple:

$$A = \langle \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty, \{\mathbf{A}_\sigma\}_{\sigma \in \Sigma} \rangle \tag{1}$$

where: $\boldsymbol{\alpha}_0$, $\boldsymbol{\alpha}_\infty \in \mathbb{R}^k$ are the initial and final weight vectors; and $\mathbf{A}_\sigma \in \mathbb{R}^{k \times k}$ are the transition matrices associated to each symbol $\sigma \in \Sigma$.

The function $f_A : \Sigma^\star \rightarrow \mathbb{R}$ realized by a WA A is defined as:

$$f(x) = \boldsymbol{\alpha}_0^\top \mathbf{A}_{x_1} \cdots \mathbf{A}_{x_n} \boldsymbol{\alpha}_\infty \quad . \tag{2}$$

Probabilistic Non-Deterministic Finite Automata (PNFA) are WFAs that compute a probabilistic distribution over strings. One can easily transform a PNFA into another automata that computes substring expectations via simple transformations of the model parameters, and the reverse is also true, see Balle et al. (2014) for details.

In this paper we will directly learn and use automata that compute expectations. To train the WFAs we will use the classical spectral learning method by described in Balle et al. (2014), using the scalability techniques by Quattoni et al. (2017).

2.2 WFA Classifier Ensemble

We will now describe how we combine class specific WFAs to build a sequence classifier. Let's assume that we have a set $L = \{1, \ldots, l\}$ of target class labels and a training set $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ of n labeled samples where $x \in \Sigma^\star$ is an input sequence and $y \in L$ is an output label. Our goal is to use D to learn a function mapping sequences to class labels, i.e. a classifier $c : \Sigma^\star \rightarrow L$.

We start by partitioning the training set D into l training sets $(d_1, \ldots d_l)$, one for each target class. Then for each training set d_l we train a corresponding WFA: $f_l(x) : \Sigma^\star \rightarrow \mathbb{R}$ using the spectral method. We can think that this model is computing an approximation of the expected number of times of observing a subsequence x from a sequence sampled from the distribution of sequences of class l. More generally, one can regard $f_l(x)$ as a real valued score that measures the compatibility between a subsequence x and a label l. Intuitively, think of x as an ngram feature.

With the scores computed by the class-specific WFAs we will build a prediction function. The idea is quite simple, we will run the scoring function over all ngrams up to a given length and aggregate the outputs to compute a single score measuring the compatibility of a sequence and a target class.

More precisely, we define a maximum ngram length parameter t. Given a sequence x we denote the set of all ngrams of x up to length t as $W_x = (w_1, \ldots, w_m)$. The aggregate prediction score is simply defined as:

$$z(x, l) = \sum_{w \in W_x} \frac{f_l(w)}{\sum_{l' \in L} f_{l'}(w)} \quad . \tag{3}$$

We can regard

$$\frac{f_l(w)}{\sum_{l' \in L} f_{l'}(w)} \tag{4}$$

as an approximation of the conditional distribution $P(l|w)$, since $f_l(w)$ is an approximation of an expectation and therefore is a non-negative score.

Given the aggregate scoring function $z(x, l)$ the prediction of the WFA ensemble is simply: $\text{argmax}_l z(x, l)$. Essentially, we are using the generative models in a *discriminative* manner.

A natural question to ask is why not to use the Naive Bayes score:

$$z(x, l) = \log P(l) + \sum_{w \in W_x} \log P(w|l) \tag{5}$$

where we approximate $P(w|l)$ by $f_l(w)$. To do so, instead of the expectation WFA, we would use a WFA that computes probabilities (which can be easily obtained from the WFA that computes expectations (Balle et al., 2014)). We have indeed

tried this approach but it performed poorly since the generative model cannot capture the discriminative ngrams of the data.

On the other hand we realized that the simple modification of using the generative models to make a discriminative prediction resulted in good performance.

3 Experiments

We conducted experiments on five sequence classification data sets:

- MR: This is a movie review data set where the task is to classify a sentence as positive or negative review. There are two classes and the average sentence length is 20. The total number of samples is 106,662 and the vocabulary size 18,765 (Pang and Lee, 2005).
- SST-2: This is a sentiment treebank, where the task is to predict a positive or negative sentiment label. There are two classes and the average sentence length is 19. The total number of samples is 9,613 and the vocabulary size 16,185 (Socher et al., 2013).
- Subj: This is a subjectivity data set were the task is to predict if a sentence is subjective or objective.There are two classes and the average sentence length is 23. The total number of samples is 10,000 and the vocabulary size 21,323 (Pang and Lee, 2004).
- TREC: This is a question classification data set. The task is to classify a question into six question types (e.g. a question about a location, a person, etc.). There are six classes and the average sentence length is 10. The total number of samples is 5,952 and the vocabulary size 9,592 (Li and Roth, 2002).
- CR: This data set contains reviews written by customers about various products. The task is to predict the review is positive or negative. There are two classes and the average sentence length is 19. The total number of samples is 3,775 and the vocabulary size 5,340 (Hu and Liu, 2004).

The WFA models have two parameters: the number of states k and the maximum window size t, both parameters were validated using a validation set. For k we tried [50, 100, 200] and for t we tested [2, 3, 4, 5].

DATA	CNN	WFA
MR	76.1	**77.3**
SST-2	**82.7**	81.6
Subj	89.6	**91.9**
TREC	**91.2**	90.1
CR	**79.8**	79.5
average	83.9	**84.1**

Table 1: Results of the WFA classifier against a baseline CNN.

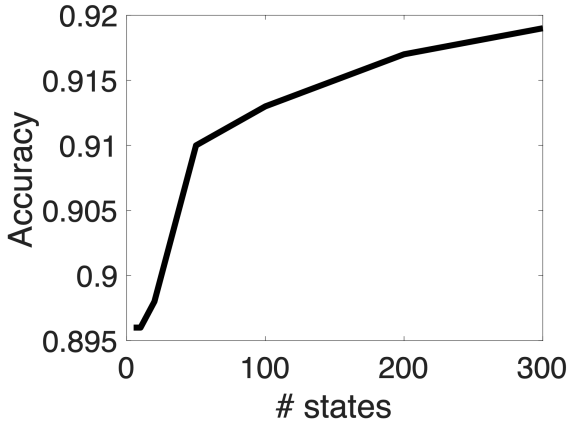

Figure 1: Performance as a function of the number of states of the model for the SUBJ dataset.

When a standard train-development-test partition was not provided in the original data set we performed 10 fold cross validation and report mean performance.

We will compare the performance of the ensemble WFA with a classical CNN architecture for sequence classification. More specifically, we compare against the model described in (Kim, 2014). The model has 100 convolutional filters run over ngrams of size: [3, 4, 5], a max-pooling layer and a fully connected softmax layer. The word embeddings are randomly initialized and then modified during training. The model was trained using 0.5 drop-out and $l2$ regularization.

We performed experiments on the 5 sequence classification datasets and report the results on Table 1. As we can see the results show that the performance of the WFA model is comparable to that of the CNN. Figure 1 shows accuracy as a function of the number of states of the model for the SUBJ dataset, as we can see even with only 5 states the model shows reasonable performance, around 89.5.

4 Discussion

In the surface CNNs and WFAs for sequence classification might seem quite dissimilar. The performance of the CNNs relies on learning good discriminative ngram features via their convolutional filters. In contrast the WFA ensemble focuses on learning good estimates of the moments distributions of each class.

However, if we look closer into the WFAs we realize that implicitly the WFA can also capture arbitrary features of the input sequences via its latent states (i.e. the latent states can remember any regular pattern of the input sequence).

The ability to induce patterns seems to be confirmed by our experiments. Our results show that by simply making a discriminative prediction out of the outputs of the class specific WFAs we can get very close to matching the performance of the CNNs.

Both CNNs and WFAs have their advantages and disadvantages. The main advantage of the CNNs is that they are very flexible and can induce arbitrary patterns. However, training them can be computationally expensive and the resulting model might be hard to interpret.

On the other hand WFAs can be easily trained with the classical spectral method. Most of the models reported in these experiments were trained in less than five minutes in a regular machine with four CPUs. In addition, because the relation between inputs and latent-state is more transparent (i.e. just a linear function) they might be interpreted more easily.

The main disadvantage of WFAs is that they lack the modeling flexibility of CNNs. This is because it is harder to incorporate arbitrary loss functions. While there have been extensions of spectral methods that can exploit any convex loss function (Quattoni et al., 2014) this usually results in optimizations that are significantly more costly. And therefore the resulting training algorithms loose part of the practical appeal of the classical spectral method.

Finally, another potential limitation of WFAs is that because the latent state dynamics is linear, they might need more states than models that can make use on non-linear dynamics.

Most likely the best model would combine the best of both worlds. But the first step is to understand their similarities and differences in the context of concrete NLP tasks. We believe our results are a first tiny step towards that goal.

5 Future Work

In many ways our experiments are crippled. The most evident limitation is that none of the models exploit external features such as pre-trained word embeddings, as it is well known that such features are essential to improve the performance of sequence prediction models.

As we already said this is just a first comparison, and we focused on the simplest possible configuration of both models. In the future, we plan to make comparisons of models that incorporate word embeddings.

Notice that WFAs have also been defined for real valued inputs (Recasens and Quattoni, 2013) and therefore they can also incorporate pre-trained word embedding vectors.

Furthermore, the comparison is relatively unfair in the sense that the WFAs are trained in a generative fashion. There have been proposals for discriminative training or discriminative refinements that we plan to explore in the future (Quattoni and Carreras, 2019; Quattoni et al., 2014).

Finally, in this paper we focus on comparisons against CNNs. But it would be interesting to expand the study to other models such as RNNs and LSTMS.

Acknowledgements

This work is supported by the European Research Council (ERC StG INTERACT 853459).

References

Raphaël Bailly, François Denis, and Liva Ralaivola. 2009. Grammatical inference as a principal component analysis problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 33–40, New York, NY, USA. ACM.

Borja Balle, Xavier Carreras, Franco M. Luque, and Ariadna Quattoni. 2014. Spectral Learning of Weighted Automata: A Forward-Backward Perspective. *Machine Learning*, 96(1):33–63.

Borja Balle, Ariadna Quattoni, and Xavier Carreras. 2011. A spectral learning algorithm for finite state transducers. In *Proceedings of the 2011th European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part I*, ECMLPKDD'11, pages 156–171, Berlin, Heidelberg. Springer-Verlag.

Shay B. Cohen, Karl Stratos, Michael Collins, Dean P. Foster, and Lyle Ungar. 2012. Spectral learning of latent-variable pcfgs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 223–231, Jeju Island, Korea. Association for Computational Linguistics.

Yoav Goldberg. 2016. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57(1):345420.

Daniel Hsu, Sham M Kakade, and Tong Zhang. 2012. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480.

Daniel J. Hsu, Sham M. Kakade, and Tong Zhang. 2009. A spectral algorithm for learning hidden markov models. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*.

M. Hu and B. Liu. 2004. Minning and summarizing customers reviews. In *Proceedings of ACL SIGKDD 2004*.

Rie Johnson and Tong Zhang. 2015. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112, Denver, Colorado. Association for Computational Linguistics.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunson. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52th Annual Meeting on Association for Computational Linguistics*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar. Association for Computational Linguistics.

X. Li and D. Roth. 2002. Learning questions classifiers. In *Proceedings of ACL 2002*.

B. Pang and L. Lee. 2004. A sentimental eduction: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*.

B. Pang and L. Lee. 2005. Seeing starts: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*.

Ariadna Quattoni, Borja Balle, Xavier Carreras, and Amir Globerson. 2014. Spectral regularization for max-margin sequence tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1710–1718. JMLR Workshop and Conference Proceedings.

Ariadna Quattoni and Xavier Carreras. 2019. Interpolated spectral NGram language models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5926–5930, Florence, Italy. Association for Computational Linguistics.

Ariadna Quattoni, Xavier Carreras, and Matthias Gallé. 2017. A maximum matching algorithm for basis selection in spectral learning. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *JMLR Proceedings*.

Guillaume Rabusseau, Tianyu Li, and Doina Precup. 2019. Connecting weighted automata and recurrent neural networks through spectral learning. In *Proceedings of Machine Learning Research*, volume 89, pages 1630–1639. PMLR.

Adria Recasens and Ariadna Quattoni. 2013. Spectral learning of sequence taggers over continuous sequences. In *Machine Learning and Knowledge Discovery in Databases*, pages 289–304. Springer Berlin Heidelberg.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng, and C. Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP 2013*.

Ye Zhang and Byron Wallace. 2017. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.