Domain Adaptation of Neural Soft Patterns for Sentiment Analysis

Ronald Cardenas Acosta
Faculty of Information and Communication Technology
University of Malta
ronald.cardenas.18@um.edu.mt

Abstract

This report describes experiments on domain adaptation for the task of sentiment analysis. We build upon the recently proposed SoPa (Schwartz et al., 2018), a hybrid CNN-RNN architecture that mimics the behaviour of a Weighted Finite State Automata. SoPa supports partial matches of end-to-end learned lexical patterns, providing an interpretable framework in which a thorough analysis of matched phrases is possible. We find that patterns inferred from a source domain can be adapted to a related target domain in order to match phrases relevant to said target domain, although with marginal gains in classification accuracy.

1 Introduction

The objective of domain adaptation techniques is to adapt a hypothesis trained on a source data distribution so that it can perform well on a related target distribution. These techniques have been applied to a variety of NLP tasks such as sentiment analysis (Blitzer et al., 2007; McAuley and Leskovec, 2013; McAuley et al., 2015; Ruder and Plank, 2018), style transfer in text generation (Fu et al., 2018; Yang et al., 2018; Peng et al., 2018), textual and visual question answering (Chao et al., 2018; Zhao and Liu, 2018), and machine translation (Etchegoyhen et al., 2018; Britz et al., 2017), to name a few.

In the case of sentiment analysis of online user reviews, previous work has sought effective ways of transfer learning between product categories (Blitzer et al., 2007; Ruder and Plank, 2018). However, the task has been proven to be challenging since sentiment is expressed differently in different domains. For instance, Blitzer et al. (2007) identifies three types of feature behaviour across domains: (a) features that are highly predictive in the source domain but not in the target domain, (b) features that are highly predictive in the target domain but not in the source domain, and (c) features that are positively predictive in the source domain but are negatively predictive in the target domain (or viceversa).

In this report, we focus on unsupervised domain adaptation for the task of sentiment analysis, transfering from a single source domain into a single target domain. We build upon the recently proposed SoPa (Schwartz et al., 2018), a neural architecture that mimic the behaviour of a Weighted Finite State Machine. SoPa is able to learn soft lexical patterns, i.e. word patterns that might include a (possibly empty) wild card. We investigate the performance of SoPa under a self-training setup following calibration procedures proposed by Ruder and Plank (2018). Experiments on Amazon online reviews of two product categores show marginal improvements over a direct transfer approach. However, adapted patterns succesfully score phrases relevant to the target domain higher than not adapted patterns. Clearly, the adaptation method shows plenty room for improvement.

2 Related Work

Early work on domain adaptation for sentiment analysis, namely non-neural approaches, reports that transfering from a source domain closer to the target domain yields better performance than combining several significantly varied domains (Blitzer et al., 2007; Aue and Gamon, 2005). One identified reason is the vocabulary mismatch between domains, leading to scenarios where features drawn from one domain are not present in the other or contradict each other, as reported by Blitzer et al. (2007). In the advent of neural networks, this problem is partially addressed with continuous representation of words. A more direct approach is taken by Barnes et al. (2018)

who projects embeddings from both source and target domains into a common space in an adversarial setup. Furthermore, most neural architectures proposed so far rely on pretrained word embeddings that could be considered domain-independent given the datasets these embededdings were trained on (Pennington et al., 2014; Peters et al., 2018). These huge benmark datasets, e.g. Wikipedia, CommonCrawl, are meant to be as varied as possible in terms of domains.

However, highly specific domains will present word types that are likely not represented in these pretrained representations. In this case, a model will rely on the embedding module's robustness to represent OOV types. In this scenario, (Schwartz et al., 2018) proposes SoPa, a model that mimics the behaviour of a Weighted Finite State Machine. The model itself can be regarded as a restricted case of a one-layer CNN that consumes the input one token at a time, like an RNN. The architecture shifts the representation robustness from the token level to the phrase level by modelling a soft version of traditional lexical patterns. The model learns to represent fixed-length patterns of words with possibly empty or extra components. For example, a soft pattern could match the sequence *A B C* as well as *A * C*.

The performance of SoPa is tested by Schwartz et al. (2018) for the task of sentiment analysis in single domain scenarios. In this report, we investigate the performance of SoPa under a transfer learning scenario from one source domain (Movies & TV) to one target domain (Games).

3 WSFAs and Soft Patterns

A soft pattern, as introduced by Davidov et al. (2010), is a pattern that supports partial matching on a given span of text by skipping some words of the pattern. Let WFSA-ϵ be a WFSA that support ϵ transitions (a transition that skips an input word) as well as self-loops (a transition that repeats the insertion of an input word). Let WFSA-ϵ be defined by the tuple $F = \langle S, V, \pi, T, \eta \rangle$ where S is the set of states with size d, V is the vocabulary, $\pi \in \mathbb{R}^d$ is the weight vector for initial states, $T : (V \cup \{\epsilon\}) \to \mathbb{R}^{d\times d}$ is a transition weight function, and $\eta \in \mathbb{R}^d$ is the weight vector for final states. Then, a sequence of word tokens $w = \langle w_0, ..., w_n \rangle$ can be scored using the Forward algorithm, as follows,

$$p(\mathbf{w}) = \pi^T T(\epsilon)^* \left(\prod_{i=1}^{n} T(w_i) T(\epsilon)^* \right) \eta \qquad (1)$$

where $T^* = \sum_{j=0}^{\infty} T^j$, which can be approximated by its first order expansion for computational reasons as $T^* \approx I + T$. By doing so, the pattern would allow only one ϵ-transition per match.

A pattern as a neural WSFA. A WFSA based on neural weights has the potential to support partial matchings for a given pattern. Let a pattern of fixed length d be instanciated by a specific transition function T which is defined as follows,

$$[T(w)]_{i,j} = \begin{cases} E(u_i \cdot v_w + a_i), & \text{if} j = i \text{ (self-loop)} \\ E(w_i \cdot v_w + b_i), & \text{if} j = i+1 \\ 0, & otherwise \end{cases} \qquad (2)$$

where u_i and w_i are weight parameters, a_i and b_i are bias terms, v_w is the embedding representation of token w, and E is an encoding function. Schwartz et al. (2018) propose the sigmoid as encoding function in order to discourage the model from following too many self loops and keep the match length as close as possible to the number of states d.

Equation 2 also presents an interesting property of the transition matrix. The tradeoff between magnitudes of w_i and b_i allows the pattern to vary the matching range from a specific word form (a large w_i and a smallb_i) to any word form (a small w_i and a large b_i).

Scoring with a pattern. Given a pattern F, a word sequence is consumed one token at a time following Equation 1. At each timestep, the pattern can choose between three possible actions: (a) transitioning to the next state and consume one token, (b) to not transition to the next state and consume a token (a self-loop), or (c) transition to the next state and not consume a token (an ϵ-transition). At each timestep, the highest scoring path though F is calculated by restricting F to the max-product semiring. Then, the final document score obtained by a given pattern is the maximum score obtained after consuming all tokens.

It is worth noting that at a given timestep, the score distribution over states depends on the current token and the previous distribution of states. In this sense, F can be considered a single-layer RNN.

Scoring with SoPa. So far we have considered only one pattern and the final score it obtains after consuming a whole document. The model proposed by (Schwartz et al., 2018), SoPa, aggregates the final scores of many patterns of different lengths into a feature layer for classification.

4 Domain Adaptation with SoPa

We resort to the bootstrapping method of throttling self-training treating the SoPa architecture as a black box. Let $\mathcal{D}_{src} = \langle X_{src}, Y_{src} \rangle$ be the dataset in the source domain, composed of documents X_{src} and their respective sentiment class labels Y_{src}. The training pipeline starts with the training of a SoPa model M on $\mathcal{D}_{src}$. Then, we proceed to self-train M on unlabeled data in the target domain, $\mathcal{D}_{tgt} = \langle X_{tgt} \rangle$. At each iteration, M provides probablity distributions over the class label set for all unlabeled documents in X_{tgt}. Following calibration procedures outlined by (Ruder and Plank, 2018), we select the top n unlabeled instances according to their confidence prediction, namely the probability provided by M. These n instances $\langle X'_{tgt}, \hat{Y}_{tgt} \rangle$ are added to the training set $\mathcal{D}_{src}$ and M is re-trained. Then, the next iteration takes place.

5 Experimental Setup

5.1 Dataset

We use the provided dataset, a balanced subset of the reviews data extracted by McAuley et al. (2015). The data consists of users reviews on two domains –Movies & TV, and Games–, extracted from Amazon. We use Movies & TV category as source domain and Games as target domain. We extract a development subset from the source domain and further divide the target domain's data into unlabeled, development, and test splits. Table 1 presents the sizes of each split considered in the experiments.

5.2 Training of source domain

We use pre-trained 300-dimensional GloVe 840B embeddings Pennington et al. (2014) normalized to unit length. Training of the SoPa model was performed using Adam (Kingma and Ba, 2014) as optimizer.

For hyper-parameter tunning, we resort to a subset of the training and development source data consisting of 10,000 and 5,000 instances, respectively. These subsets were sampled without replacement following a uniform distribution. We use a Tree-structured Parzen Estimator (TPE) optimization model over 30 iterations[1]. Table 2 shows the range of hyper-parameter values explored and the optimal values found.

5.3 Domain adaptation models

We take as baseline a model trained only over the source domain data. This model is used to obtain predictions in the target domain without any sort of adaptation, i.e. under a direct transfer approach. We call this model M_{src}.

Then, we experiment with throttling self-training as a domain adaptation technique. Preliminary experiments showed that choosing the top 80% most confident prediction in each self-training iteration yielded the best results. We self-train for 3 iterations, each iteration training the model over $\mathcal{D}_{src} \cup \langle X'_{tgt}, \hat{Y}_{tgt} \rangle$ for 10 epochs. The resulting model is called M_{tgt}.

5.4 Interpretability analysis

Following Schwartz et al. (2018)'s work, we isolate patterns inferred by both SoPa models, M_{src} and M_{tgt}, and analyse their contribution to the classification task. The final linear layer in the architecture of SoPa allows us to directly analize the contribution of each pattern's final score for certain document. This contribution is defined as the difference in accuracy after zeroing out the score of a pattern under a leave-one-out setup.

6 Results and Discussion

6.1 Domain adaptation

For comparison purposes, we report results of the baseline model over the source domain. This model, M_{src}, which was tuned and optimized over the source domain, obtains 82.02% of accuracy over the source domain test set.

Table 3 presents results on the target domain before and after adaptation of the baseline. It can observed that a self-training approach (M_{tgt}) marginally improves over a direct transfer approach (M_src). This behaviour can be attributed to the limited amount of unlabeled data and few iterations of self-training performed.

Our pipeline could also benefit from improvements orthogonal to the architecture itself, such as the usage of more contextualized pre-trained word

[1] We use HyperOpt library (http://hyperopt.github.io/hyperopt/)

Domain	Train	Dev	Test	Unlabeled
Movies & TV (src)	89,998	17,999	10,000	-
Games (tgt)	-	5,000	11,142	5,000

Table 1: Size of data splits in source (src) and target (tgt) domains.

Hyper-parameter	Range	Optimal
Patterns	{6:10, 5:10, 4:10, 3:10, 2:10}, {6:10, 5:10, 4:10}	{6:10, 5:10, 4:10}
Learning rate	10^{-9}–10^{-2}	0.00015
Dropout	0–0.2	0.0017
MLP hid. dim.	100–300	100
Batch size	10–64	20

Table 2: Range and optimal values of hyper-parameters tuned over source domain data.

	dev	test
M_{src}	80.20	80.35
M_{tgt}	80.76	80.65

Table 3: Sentiment analizis results on the target domain *Games* of SoPa model trained only on source domain data (M_{src}) and self-trained SoPa model (M_{tgt}).

embeddings (Peters et al., 2018; Devlin et al., 2018).

6.2 Interpretability of patterns

We analize the interpretability of the patterns inferred by the adapted model, M_{tgt}. Table 5 presents the top scored phrases in the development set, grouped by length, along with the gold label associated with the document these phrases appear in. We observe that negative phrases, such as *rudely dissapointed* and *so lame*, are correctly associated with negative sentiments (label 0). Analogously, positive gaming–related phrases, such as *multiplayer capability* and *ps-2 memorabilia*, are associated with positive sentiments (label 1).

It is also worth noting the preference of the model to include ϵ-transitions in order to match shorter relevant phrases.

6.3 Interpretability of predictions

We futther analize the interpretability of predictions by inspecting the top scoring patterns for a certain document. Table 4 presents a sample instance in which the baseline fails to predict the correct label but the adapted model correctly predicts it. The sample text talks about an NFL game and mentions several –back then– famous players' names such as *tom hammond* and *chris collinsworth*.

On one hand, we observe that patterns inferred by M_{src} weight proper names highly, probably because a movie review is most likely to mention names of actors and actresses. On the other hand, we observe that M_{tgt} managed to diversify the matching space of the inferred patterns. For example, the top contributing pattern includes the version of the game (“*’10*”) in addition to the player's name. Furthermore, a phrase relevant to gaming is now highly scored (“*new features*”).

Here as well, we observe the soft nature of the patterns inferred, indicated by the presence of ϵ-transitions in all observed patterns.

7 Conclusion

We investigate the behaviour of SoPa, a neural architecture with WFST-like inference recently proposed by Schwartz et al. (2018) for the task of sentiment analysis of online user reviews under domain shift. We experiment with a self-training method for domain adaptation considering calibration procedures suitable for neural networks. The soft nature of the patterns inferred by SoPa, parameterized by their transition matrixes, provides an interpretable framework suitable to analyze how pattern scoring changes after adapting to another domain. When transfering from a related target domain (from Movies & TV to Games) we obtain an improvement of classification accuracy, although marginal. However, under closer inspection, we observe that the adapted patterns match phrases highly relevant to the target domain.

Text	i just got madden 10 and also have ’08 , so i will be comparing this to those two ...				
	... plus the updated rosters and new features on superstar mode ...				
	... ’10 has chris collinsworth and tom hammond ...				
	... i really do not like tom hammond ...				
	... i think it is better , and completely worth it .				
Gold label	1				
Model	M_{src}				
Prediction	0				
Top contributing patterns (score)	(0.263)	ϵ	and	tom	hammond
	(0.263)	ϵ	chris	ϵ	collinsworth
	(0.263)	ϵ	tom	hammond	
Model	M_{tgt}				
Prediction	1				
Top contributing patterns (score)	(0.668)	’10	has	chris	ϵ	collinsworth
	(0.667)	ϵ	tom	ϵ	hammond
	(0.647)	ϵ	new	features	

Table 4: Text and gold label of sample instance from the development set (top row), prediction and top contributing patterns according to the baseline (middle row) and domain–adapted model (bottom row). ϵ: ϵ-transition.

Length of pattern	Top scoring phrases					Gold label
4	multplayer	capability	ϵ	(		1
	dissapointed	by	ϵ	it		0
	ϵ	gameplay	!	there		1
	suggest	looking	around	for		0
	a	ϵ	ps-2	memorabilia		1
5	ϵ	’s	just	so	lame	0
	ϵ	rudely	dissapointed	by	it	0
	no	ϵ	more	of	that	1
	great	ϵ	idea	:	take	0
	ϵ	multplayer	ϵ	capability	(	1

Table 5: Top scoring phrases and the gold labels of the document they appear in the development set according to patterns of length 4 and 5 (one pattern per row).

References

Anthony Aue and Michael Gamon. 2005. Customizing sentiment classifiers to new domains: A case study. In *Proceedings of recent advances in natural language processing (RANLP)*, volume 1, pages 2–1. Citeseer.

Jeremy Barnes, Roman Klinger, and Sabine Schulte im Walde. 2018. Projecting embeddings for domain adaption: Joint modeling of sentiment analysis in diverse domains. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 818–830.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Denny Britz, Quoc Le, and Reid Pryzant. 2017. Effective domain mixing for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 118–126.

Wei-Lun Chao, Hexiang Hu, and Fei Sha. 2018. Cross-dataset adaptation for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5716–5725.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd international conference on computational linguistics: posters*, pages 241–249. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Thierry Etchegoyhen, Anna Fernández Torné, Andoni Azpeitia, Eva Martínez Garcia, and Anna Matamala. 2018. Evaluating domain adaptation for machine translation across scenarios. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52. ACM.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Sebastian Ruder and Barbara Plank. 2018. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1044–1054.

Roy Schwartz, Sam Thomson, and Noah A Smith. 2018. Sopa: Bridging cnns, rnns, and weighted finite-state machines. In *Proceedings of ACL*.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised Text Style Transfer using Language Models as Discriminators. In S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7287–7298. Curran Associates, Inc.

Helen Jiahe Zhao and Jiamou Liu. 2018. Finding answers from the word of god: Domain adaptation for neural networks in biblical question answering. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.