Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Transition-based Adversarial Network for Cross-lingual Aspect Extraction

Wenya Wang[†‡] and **Sinno Jialin Pan**[†]
[†]Nanyang Technological University, Singapore
[‡]SAP Innovation Center, Singapore
{wa0001ya, sinnopan}@ntu.edu.sg

Abstract

In fine-grained opinion mining, the task of aspect extraction involves the identification of explicit product features in customer reviews. This task has been widely studied in some major languages, e.g., English, but was seldom addressed in other minor languages due to the lack of annotated corpus. To solve it, we develop a novel deep model to transfer knowledge from a source language with labeled training data to a target language without any annotations. Different from cross-lingual sentiment classification, aspect extraction across languages requires more fine-grained adaptation. To this end, we utilize transition-based mechanism that reads a word each time and forms a series of configurations that represent the status of the whole sentence. We represent each configuration as a continuous feature vector and align these representations from different languages into a shared space through an adversarial network. In addition, syntactic structures are also integrated into the deep model to achieve more syntactically-sensitive adaptations. The proposed method is end-to-end and achieves state-of-the-art performance on English, French and Spanish restaurant review datasets.

1 Introduction

Different from coarse-grained sentiment classification which predicts an overall sentiment polarity for each sentence, fine-grained opinion mining involves the identification of aspect terms describing product features, which is important for information extraction. However, for this task, a crucial issue is the difficulty in collecting sufficient labeled data to train a precise classifier, especially for resource-limited languages. Existing cross-lingual approaches mainly focused on coarse-grained sentiment classification. To the best of our knowledge, there is yet no studies on cross-lingual aspect extraction, due to the difficulties in word-level feature adaptation.

Aspect extraction in supervised mono-lingual setting has been studied extensively [Qiu *et al.*, 2011; Liu *et al.*, 2015; Yin *et al.*, 2016; Wang *et al.*, 2016; 2017]. Given a sentence e.g., "We love the pink pony and atmosphere.", the aspect terms to be extracted are *pink pony* and *atmosphere*. Most existing work treat this task as a sequence labeling problem that predicts a label for each token. It has also been shown that syntactic relations are crucial to identify target terms [Qiu *et al.*, 2011; Yin *et al.*, 2016; Wang *et al.*, 2016]. Token-level knowledge transfer has been proposed for cross-domain aspect extraction [Li *et al.*, 2012; Ding *et al.*, 2017], where the models are built upon common sentiment lexicons. However, these methods only work for homogeneous space within the same language. Moreover, token-level knowledge is much more difficult to be transferred across languages, because the segmentation rules may vary in different languages. For instance, an aspect can be expressed by a single word in one language but by multiple words in another language. To conquer this limitation, we adopt transition-based models, which have been used for dependency parsing [Dyer *et al.*, 2015] and named entity recognition [Lample *et al.*, 2016]. Different from sequence labeling, transition-based models learn a sequence of actions that read words sequentially from a buffer and avoid the separation of a phrase. These actions depend on the configuration representation at the previous timestamp. To transfer knowledge, we aim to learn a shared space for configurations from different languages.

Recently, adversarial networks [Goodfellow *et al.*, 2014] have been widely used and shown promising results for domain adaptation [Ganin *et al.*, 2016]. Chen *et al.* [2016] applied the domain adversarial network to cross-lingual sentiment classification which learns a domain discriminator for sentence representations. However, fine-grained adaptation is much more difficult than sharing of sentence representations. To make adaptation successful, we propose a transition-based adversarial network, where a generator is used to produce language-invariant configuration features, and a discriminator competes to discriminate the configurations between source and target languages. Since dependency relations have proven to play an important role in identification of aspect terms, and are invariant across different languages when using universal dependency parsers [de Marneffe *et al.*, 2014], we build partial dependency trees when computing the configuration representations to capture the key components during transfer. We also use these relations as auxiliary labels that helps the adversarial network to learn better representations. Our main contributions are three-fold: 1) We propose a novel transfer strategy that uses sentence configuration as

invariant features for cross-lingual word predictions. 2) We integrate syntactic information to assist knowledge transfer across different languages. 3) We conduct extensive experiments to show the effectiveness of our proposed model.

2 Related Work

Aspect extraction has been actively studied for English corpus. Early works have applied unsupervised methods to extract the target words using association rules mining [Hu and Liu, 2004] and manually designed rules based on syntactic relations [Qiu *et al.*, 2011]. Several supervised approaches have later been proposed including HMMs with feature engineering [Jin and Ho, 2009] and deep learning [Liu *et al.*, 2015; Yin *et al.*, 2016; Wang *et al.*, 2016]. These supervised methods all treat the problem as a sequence labeling task that work on each token in a sentence. For cross-domain aspect extraction, Li *et al.* [2012] proposed a boosting-style method to expand target lexicon through common sentiment words and syntactic relations across domains. Ding *et al.* [2017] developed a deep model that considers structural correspondences. The above methods depend on common sentiment lexicon, which is not available for heterogeneous spaces across languages.

For cross-lingual sentiment classification, one approach is to use machine translation to translate into source or target language to build a classifier for target language [Wan, 2009]. Another approach is to use parallel corpora or word pairs to learn shared features [Prettenhofer and Stein, 2010; Meng *et al.*, 2012; Zhou *et al.*, 2015]. Bilingual representation learning has also been widely studied [Klementiev *et al.*, 2012; Hermann and Blunsom, 2014; Chandar *et al.*, 2014; Gouws *et al.*, 2015; Zhou *et al.*, 2016]. Moreover, Zhou *et al.* [2014] used marginalized stack denoise auto-encoder to align document representations from heterogeneous spaces. Chen *et al.* [2016] applied domain adversarial network [Ganin *et al.*, 2016] to train language-independent document representations. Among all these, fine-grained-level adaptation has not been investigated yet.

3 Problem Definition & Motivation

The task studied in this paper involves the identification of explicit aspect terms in each review sentence. Formally, given a sequence of words $\mathbf{x} = (w_1, w_2, ..., w_n)$, the transition model reads words sequentially and outputs a sequence of actions $\mathbf{y} = (y_1, y_2, ..., y_m)$, resulting in a sequence of configurations $\mathbf{c} = (c_1, c_2, ..., c_m)$, where m is the number of timestamps. Each $y_t \in A$, where $A = \{\text{OUTPUT}, \text{SHIFT}, \text{REDUCE}\}$. Each $c_t = (c_t^o, c_t^c, c_t^b)$ consists of three components that partition the whole sentence: output stack (c_t^o), candidate stack (c_t^c) and buffer (c_t^b), where each component consists of a subsequence of tokens (or empty). The aspect terms are the complete segments in c_t^c when $y_{t+1} = \text{REDUCE}$.

In transition-based models, each configuration c_t represents a partition of the whole sentence at timestamp t. The concept of configuration is our primary motivation for knowledge transfer across different languages. Since each configuration summarizes the current status and decides on the next action, by aligning them from different languages into a shared space, we can transfer the knowledge of how to take actions in the target language. Compared with token-level representation, the configuration represents the global information including the entire history processed and the future ahead. The adaptation within a transition system is more aligned with cognitive modeling: a reader makes incremental actions in the same way regardless of different languages. Moreover, token-level predictions require accurate learning for different positions within word segments, e.g., the prediction for *service staff* should be "B" (beginning of aspect) for *service* and "I" (inside of aspect) for *staff*. However, the translation is *personnel de service* in French, which consists of 3 words. When the segments differ, adaptation becomes noisy and inaccurate. This could be addressed using transition systems, which use a candidate stack to store and represent a complete aspect segment to be transfered across languages.

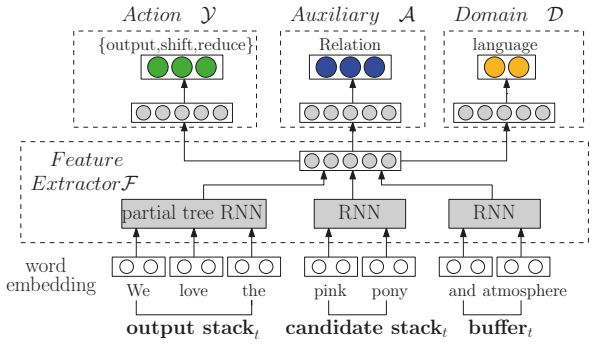


Figure 1: The overall architecture TAN.

It has been shown that universal dependency relations are invariant across different languages. Since Dyer *et al.* [2015] showed the effectiveness of a transition model for dependency parsing by predicting the transition or relation at each timestamp according to the current configuration, we take this sequential relation prediction as an auxiliary task to our main task. Moreover, motivated by auxiliary conditioned generative adversarial network (ACGAN) [Odena *et al.*, 2017], we integrate this auxiliary task into the transition-based adversarial network to make the transfer more coherent with the language-invariant syntactic structure. In addition, we integrate syntactic structure into the representation of each configuration. This helps to make the representations more sensitive to the important words in the sentence. For example, if the stack consists of "We love", the dependency tree will output the representation for *love*, which is the parent of *We*. This construction is able to attend on the important word *love*.

4 The Proposed Model

We name our proposed model **T**ransition-based **A**dversarial **N**etwork (TAN) in the sequel. Figure 1 shows the overall structure which consists of a feature extractor ($\mathcal{F}$) and three classifiers: an action classifier ($\mathcal{Y}$), an auxiliary predictor ($\mathcal{A}$) and a domain discriminator ($\mathcal{D}$). At timestamp t, the input sequence is partitioned into an output stack, a candidate stack and a buffer. Each component computes its own representation through a recurrent neural network (RNN) in the last layer, which are then integrated into the configuration rep-

Action	Output	Candidate	Buffer	Segment
	[]	[]	[We love the pink pony and atmosphere]	—
OUTPUT	[We]	[]	[love the pink pony and atmosphere]	—
OUTPUT	[We love]	[]	[the pink pony and atmosphere]	—
OUTPUT	[We love the]	[]	[pink pony and atmosphere]	—
SHIFT	[We love the]	[pink]	[pony and atmosphere]	—
SHIFT	[We love the]	[pink pony]	[and atmosphere]	—
REDUCE	[We love the pink pony]	[]	[and atmosphere]	(pink pony)
OUTPUT	[We love the pink pony and]	[]	[atmosphere]	—
SHIFT	[We love the pink pony and]	[atmosphere]	[]	—
REDUCE	[We love the pink pony and atmosphere]	[]	[]	(atmosphere)

Figure 2: An example of all the transition actions and states of each configuration of a sentence.

resentation that is fed into all three classifiers. In the following, we first introduce a basic transition-based model in Section 4.1, then illustrate the integration of an adversarial component in Section 4.2, and finally present the final model by considering the syntactic information in Section 4.3.

4.1 Transition-based Model

The transition-based method is formalized as a sequence of actions that read words sequentially from a buffer [Dyer *et al.*, 2015]. It consists of an output stack, a candidate stack and a buffer. The output stack stores the processed words, the candidate stack stores incomplete aspect candidates until all the words for an aspect are processed. The buffer contains the words that have yet to be processed. There are three possible transition actions: Output, Shift, and Reduce.

1. **Output** - Move the word on top of the buffer to the output stack and label it as "None".
2. **Shift** - Move the word on top of the buffer to the candidate stack.
3. **Reduce** - Move every word in the candidate stack to the output stack and label them as "Aspect".

The algorithm starts with the buffer containing every word in the sentence. At each configuration, one action is taken, until the algorithm terminates when both the candidate stack and buffer are empty. An illustration of the process is shown in Figure 2. The next action depends on the current configuration, which is computed from the representations of the 3 components: output stack, candidate stack and buffer. We use Gated Recurrent Unit (GRU), one of the RNN architectures to compute the representation of each component. A GRU takes the word embedding as the input and produces the output of the last cell to represent the corresponding subsequence, as shown in Figure 3 on the left. We take the reverse GRU to compute the buffer representation and use another vector to represent empty stack when the candidate stack is empty.

Formally, let $\mathbf{x}_t^O$, $\mathbf{x}_t^C$ and $\mathbf{x}_t^B$ denote the input sequence for output stack, candidate stack and buffer, respectively, at timestamp t. Denote the hidden representations by $\mathbf{h}_t^O$ for output stack, $\mathbf{h}_t^C$ for candidate stack, and $\mathbf{h}_t^B$ for buffer. The final feature representation for the configuration is

$$\mathbf{h}_t = \tanh(\mathbf{W}_h[\mathbf{h}_t^O : \mathbf{h}_t^C : \mathbf{h}_t^B] + \mathbf{b}_h), \quad (1)$$

where $\mathbf{h}_t^O = f_{GRU}(\mathbf{x}_t^O, h_0; \Theta)$, $\mathbf{h}_t^C = f_{GRU}(\mathbf{x}_t^C, h_0; \Theta)$, and $\mathbf{h}_t^B = f_{RGRU}(\mathbf{x}_t^B, h_0; \Theta)$. Here, the operator [:] denotes the concatenation of vectors, the model f_{GRU} produces the last hidden representation of a forward GRU network, and the model f_{RGRU} indicates the reverse GRU network that reads words in the backward direction. We use Θ to denote all the GRU parameters. The action $\mathbf{y}_{t+1}$ to be taken at the next time stamp is obtained via $\mathbf{y}_{t+1} = \text{softmax}(\mathbf{W}_y\mathbf{h}_t + \mathbf{b}_y)$.

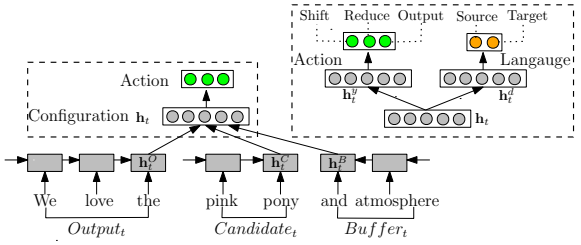


Figure 3: Configuration representation constructed from RNNs.

4.2 Transition-based Adversarial Network

To adapt the above model to another language, we apply domain adversarial network (DAN) to learn domain-invariant features [Ganin *et al.*, 2016]. The intuition is to make each configuration representation invariant across languages, but discriminative for action predictions. Indeed, at each state of transition, the configuration should share similar characteristics across languages to inform the next action. To apply DAN, we create two classifiers: a transition action predictor $\mathcal{Y}$ and a language discriminator $\mathcal{D}$ to discriminate between configurations from source and target languages.

The corresponding structure is shown in Figure 3 on the right. Denote by θ_f, θ_y, and θ_d the parameters for feature learning, action prediction, and language prediction, respectively. The discriminator generates a probability distribution over languages $P(S|\mathbf{h}_t)$, where $\mathbf{h}_t$ obtained from (1) is the hidden representation for the whole configuration at time stamp t. Here $P(S = 1|\mathbf{h}_t)$ indicates the probability of $\mathbf{h}_t$ belonging to the source language. The predictions for action label and language label are computed on top of a fully connected layer

$$\mathbf{y}_{t+1} = \text{softmax}(\mathbf{W}_y\mathbf{h}_t^y + \mathbf{b}_y), \quad (2)$$

$$\mathbf{y}_t^d = \text{softmax}(\mathbf{W}_d\mathbf{h}_t^d + \mathbf{b}_d), \quad (3)$$

where $\mathbf{h}_t^y = \tanh(\mathbf{W}_h^y\mathbf{h}_t + \mathbf{b}_h^y)$ and $\mathbf{h}_t^d = \tanh(\mathbf{W}_h^d\mathbf{h}_t + \mathbf{b}_h^d)$. The model is trained by minimizing

$$\mathcal{E}_1(\theta_f, \theta_y, \theta_d) = \mathcal{L}_y(\theta_f, \theta_y) - \alpha\mathcal{L}_d(\theta_f, \theta_d) \quad (4)$$

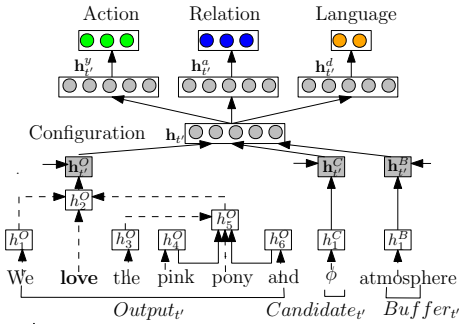


Figure 4: A configuration with proposed transition-based model.

for feature generator, and by minimizing $\mathcal{L}_d(\theta_f, \theta_d)$ for the language discriminator. Here $\mathcal{L}_y$ and $\mathcal{L}_d$ are the cross-entropy losses for action prediction and language classifier, respectively. The hyper parameter α is used to tune the trade-off between the two losses. The saddle points $\hat{\theta}_f$, $\hat{\theta}_y$, and $\hat{\theta}_d$ can be found as a stationary point of the following updates:

$$\theta_f \leftarrow \theta_f - \mu\left(\frac{\partial \mathcal{L}_y}{\partial \theta_f} - \alpha\frac{\partial \mathcal{L}_d}{\partial \theta_f}\right), \quad (5)$$

$$\theta_y \leftarrow \theta_y - \mu\frac{\partial \mathcal{L}_y}{\partial \theta_y}, \quad (6)$$

$$\theta_d \leftarrow \theta_d - \mu\left(\alpha\frac{\partial \mathcal{L}_d}{\partial \theta_d}\right). \quad (7)$$

4.3 Incorporate Syntactic Relations

The previous model ignores the syntactic relations among the words in each sentence, which is, however, crucial for aspect extraction and bridging the gap between source and target languages. To incorporate syntactic information in monolingual setting, Wang *et al.* [2016] proposed a dependency-tree-based recursive neural network that computes the hidden representation of each word as a composition function of its children. Following this idea, a straightforward method is to obtain the hidden representation of each word through recursive neural network and use it as the input to GRU for the output stack, candidate stack and buffer. However, the relation information may have little effect to the action prediction when the dependency path is too long and the model consists of multiple layers. Moreover, the hidden representation for each stack/buffer may contain information from the others due to the dependency tree. This contradicts the idea that each stack only contains partial information given by its content. To overcome these potential limitations, we propose to construct partial trees in the output stack.

Figure 4 presents an example configuration at timestamp t'. For each sentence, we first produce a dependency tree from the parser. At each configuration, we pick all the edges in the tree that connect the words in the output stack and keep the representation for the highest node in each connected component. The rest of the words in the output stack that are not connected are then linked through recurrent edges. As shown in Figure 4, the output stack consists of "We love the pink pony and". According to the dependency tree, *love* is the highest node and is connected to all the other words in the output stack, hence, we only keep the hidden representation of *love* that is recursively computed from all the words below. Formally, given an input sentence with pre-generated dependency tree, at time stamp t, let $\mathbf{x}_t^O = \{x_1^O, ..., x_{n_t^O}^O\}$, $\mathbf{x}_t^C = \{x_1^C, ..., x_{n_t^C}^C\}$, $\mathbf{x}_t^B = \{x_1^B, ..., x_{n_t^B}^B\}$ represents the list of word embeddings for the output stack, candidate stack, and buffer, respectively. For the output stack, we compute the hidden representation h_i^O of each word as:

$$h_i^O = \tanh\left(\mathbf{W}_v \cdot x_i^O + \mathbf{b} + \sum_{k \in \mathcal{K}_i^O} \mathbf{W}_{r_{ik}} \cdot h_k^O\right), \quad (8)$$

where $\mathcal{K}_i^O$ denotes the set of children of node i in the output stack, r_{ik} denotes the dependency relation between node i and its child node k, and h_k^O is the hidden vector of node k in the output stack. When $\mathcal{K}_i^O = \emptyset$, we ignore the last addition in (8). To compute the final hidden representation $\mathbf{h}_t^O$ for output stack, we select all the h_j^O's when the jth node does not have any parent in the output stack or it is the root node in the tree and denote the set of these node indexes as $\mathcal{I}_t^O$:

$$\mathbf{h}_t^O = f_{GRU}(\{h_j^O\}, h_0; \Theta), \quad (j \in \mathcal{I}_t^O). \quad (9)$$

The hidden representations for the candidate stack and the buffer are produced by GRU and RGRU, respectively, as

$$\mathbf{h}_t^C = f_{GRU}(\{h_i^C\}, h_0; \Theta), \ (i \in \{1, ..., n_t^C\}), \quad (10)$$

$$\mathbf{h}_t^B = f_{RGRU}(\{h_i^B\}, h_0; \Theta), \ (i \in \{1, ..., n_t^B\}), \quad (11)$$

where $h_i^C = \tanh(\mathbf{W}_v \cdot x_i^C + \mathbf{b})$ and $h_i^B = \tanh(\mathbf{W}_v \cdot x_i^B + \mathbf{b})$. We only incorporated the syntactic information in $\mathbf{h}_t^O$ because only the output stack contains rich processed information. And we use another embedding to represent the empty stack, which is randomly initialized and trained.

To explicitly make use of the syntactic relations, we integrate an auxiliary task to predict the dependency relation of the word about to be processed in each configuration. Inspired by [Odena *et al.*, 2017] which applies an auxiliary classifier to help the adversarial training, our auxiliary task to predict the relation between the word on top of the buffer and its parent is helpful to decide on next action. Moreover, this implicitly conditions the language representations $\mathbf{h}_t^d$ on the relation labels, which helps to better align language features based on syntactic structure. To avoid syntactic inconsistencies across languages, we apply universal dependency parsers and utilize its taxonomy[1] to group relations into more general categories as the auxiliary labels[2]. The auxiliary prediction is

$$\mathbf{y}_t^a = \text{softmax}(\mathbf{W}_a \mathbf{h}_t^a + \mathbf{b}_a), \text{ where } \mathbf{h}_t^a = \tanh(\mathbf{W}_h^a \mathbf{h}_t + \mathbf{b}_h^a).$$

By plugging the loss of the auxiliary prediction task into (4), our final objective becomes

$$\mathcal{E}_2(\theta_f, \theta_y, \theta_a, \theta_d) = \mathcal{L}_y(\theta_f, \theta_y) + \beta\mathcal{L}_a(\theta_f, \theta_a) - \alpha\mathcal{L}_d(\theta_f, \theta_d),$$

[1] http://universaldependencies.org/

[2] There are in total 15 categories. The groupings are: {nsubj, obj, iobj, nsubjpass, dobj}, {csubj, ccomp, xcomp}, {cc, conj}, {fixed, flat, compound, mwe}, {expl, obl, vocative, dislocated}, {advcl}, {advmod, discourse, neg}, {aux, cop, mark, auxpass}, {nmod, appos, nummod}, {acl}, {amod}, {det, clf, case}, {list, parataxis}, {root}, {punct, dep}

Data	Language	Training	Test	Total
En	English	2,000	676	2,676
Fr	French	1,733	696	2,429
Es	Spanish	2,070	881	2,951

Table 1: Dataset statistics showing number of sentences

where $\mathcal{L}_a$ is the cross-entropy loss for the auxiliary task with θ_a denoting its parameters. θ_y and θ_d are updated using (6)-(7). The updates for θ_f and θ_a are:

$$\theta_f \leftarrow \theta_f - \mu\left(\frac{\partial \mathcal{L}_y}{\partial \theta_f} + \beta\frac{\partial \mathcal{L}_a}{\partial \theta_f} - \alpha\frac{\partial \mathcal{L}_d}{\partial \theta_f}\right), \quad (12)$$

$$\theta_a \leftarrow \theta_a - \mu\left(\beta\frac{\partial \mathcal{L}_a}{\partial \theta_a}\right). \quad (13)$$

During training, each sentence is fed into the common feature extractor to produce configuration vectors. For source language, each configuration is fed into the action, auxiliary and language predictors, whereas for target language, only auxiliary and language classifiers are trained.

5 Experiments

For experiments, we use the restaurant reviews from English, French and Spanish taken from SemEval Challenge 2016 task 5. The statistics of the datasets are listed in Table 1. We use labeled training data from the source language and unlabled training data from the target language to train the model. For testing, we conduct both transductive and inductive experiments to test our model on the unlabeled training data and the test data of the target language. The results are shown in F1 scores. For aspect extraction, only exact match is counted as correct when dealing with multi-word aspect terms.

Regarding experimental setting, the word embeddings are pre-trained using multivec [Bérard *et al.*, 2016] that generates bilingual word embeddings with parallel corpus. The parallel corpus are from Europarl[3] that contains 2M sentences for each language. The dependency trees are generated from Stanford universal dependencies[4]. The whole network is trained with SGD using learning rate 0.02. The trade-off parameters are $\alpha = 0.1$ and $\beta = 1$. The size of word embeddings is 100 and the hidden layer size is 50. Each experiment is trained for 20 epochs and the best performance is reported. For the overall complexity, our model takes 10mins for training 1 epoch with 4000 sentences using Intel(R) Xeon(R) CPU E5-1650 v2 @ 3.50GHz. Testing is within seconds.

5.1 Experimental Results

To the best of our knowledge, there is not yet any work on cross-lingual aspect extraction. To make fair comparisons, we revise some popular models listed in the following to be fitted into the cross-lingual setting as our baselines:

- **Translate-TAN, Translate-CRF**: Translate the target data to source language using google translator, and train a TAN or linear-chain CRF with labeled source data to make predictions on the translated target data.
- **NoAdp**: Train the model only on source data using TAN without adaptation and then test on target data.
- **A-RNN**: With a RNN [Liu *et al.*, 2015] producing a hidden representation for each word, an adversarial network with a language discriminator is applied on top.
- **A-R^{2}NN**: Apply the model in [Wang *et al.*, 2016] but replace CRF with RNN on top of the dependency-tree-based recursive neural network. An adversarial network with a language discriminator is then applied.
- **CrossCRF**: Linear-chain CRF [Jakob and Gurevych, 2010] with non-lexical features that are similar across domains. In cross-lingual setting, we use universal POS tags, sentiment lexicons, universal dependencies etc.
- **CL-DSCL**: A deep model with structural correspondence learning proposed by [Ding *et al.*, 2017]. We convert the dependencies in the rules to universal dependencies and use sentiment lexicons for all three languages.

Specifically, for A-RNN, A-R^{2}NN and CL-DSCL, we combine training sentences from labeled source language and unlabeled target language to train the models. Given a training sentence, A-RNN and A-R^{2}NN compute final features for each word with RNN, which is then fed into a language discriminator to predict the language label. For source language, a token classifier is jointly applied to predict "BIO" labels. CL-DSCL integrates auxiliary classifiers into RNN. If a training sentence is from source language, both the auxiliary and "BIO" classifier are applied on top of the hidden vector of each word. Only auxiliary classifier is applied for target training sentences. The auxiliary task is to decide whether the word satisfies some rules describing universal dependency relations among aspect and opinion words. CrossCRF only takes training sentences from source language to train. All the trained models are tested on test sentences for target language to obtain "BIO" label for each token.

The comparison results are shown in Table 2. Obviously, the proposed model TAN achieves the state-of-the-art results most of the time with large performance gain. The performances of both A-RNN and A-R^{2}NN are much worse compared to TAN, even though the syntactic structure is incorporated to bridge the gap between source and target languages (A-R^{2}NN). This proves our assumption that word-level knowledge is hard to be transferred across languages. For CL-DSCL, the result shows 18.54%, 5.51% drop compared to TAN for inductive experiments transferring from English to French and Spanish, respectively. The inferior performances for the cross-domain models indicate the difficulty to build the correspondences with heterogeneous spaces across languages, compared to domain adaptation in the same language. This shows the advantage of our proposed model that learns transferable actions across domains.

We also conduct experiments to show the effect of each component of TAN. The results are listed in Table 3. The first column indicates the modification of TAN: *-hier* replaces the hierarchical grouping of dependency relations with exact relation as the label; *-aux* removes the auxiliary task; *-dan* removes domain-adversarial network; *-ptree* removes partial tree for the output stack and *+ftree* use the complete dependency tree to compute the hidden representation of each word

[3]http://www.statmt.org/europarl/

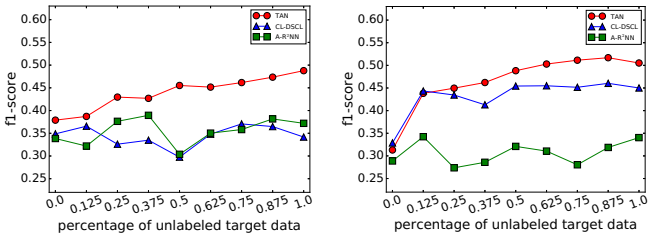
[4]https://nlp.stanford.edu/software/stanford-dependencies.shtml

Models	En→Fr		En→Es		Fr→En		Fr→Es		Es→En		Es→Fr	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Translate-TAN	45.09	40.74	45.85	41.08	39.28	38.74	32.27	34.54	45.94	41.28	41.52	36.38
Translate-CRF	25.23	23.15	28.26	30.10	25.89	26.79	31.55	30.63	32.24	26.66	24.05	20.90
NoAdp	27.71	26.13	27.56	31.31	41.21	38.29	45.43	48.21	37.52	30.39	37.95	37.89
A-RNN	22.92	20.54	31.11	34.04	29.62	27.11	40.58	40.77	35.49	30.26	34.52	31.02
A-R^{2}NN	27.92	23.41	28.63	28.65	36.43	33.25	38.55	39.45	40.83	34.16	42.83	37.19
CrossCRF	20.41	16.83	16.17	18.22	21.63	19.02	6.90	6.81	10.13	8.28	12.01	10.24
CL-DSCL	33.67	31.48	44.56	45.01	51.75	47.27	53.23	55.89	50.22	**45.90**	38.66	34.17
TAN	**53.27**	**50.02**	**49.38**	**50.52**	**55.38**	**50.30**	**55.32**	**57.65**	**51.99**	44.14	**51.16**	**48.78**

Table 2: Comparisons with different baselines.

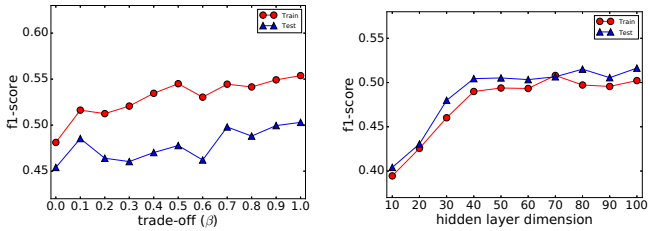
Models	En→Fr		En→Es		Fr→En		Fr→Es		Es→En		Es→Fr	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
TAN	**53.27**	**50.02**	49.38	**50.52**	**55.38**	**50.30**	**55.32**	**57.65**	**51.99**	**44.14**	**51.16**	**48.78**
-hier	50.47	45.73	**49.99**	50.23	50.70	46.48	50.54	54.07	48.48	42.28	49.81	45.89
-aux	51.30	45.05	47.22	49.00	48.12	45.40	45.17	50.29	44.01	39.05	45.91	42.47
-dan	40.04	36.20	39.87	44.09	47.91	40.40	50.18	52.83	44.44	39.56	42.80	36.04
-ptree	32.62	29.50	28.84	30.94	40.40	35.12	39.73	44.89	32.39	28.75	39.05	35.00
+ftree	48.78	43.28	47.24	48.20	52.68	48.90	50.89	53.50	49.19	43.99	50.65	45.12

Table 3: Comparisons with different variants of the proposed model.



(a) Es→Fr (b) En→Es

Figure 5: F1 score vs percentage of unlabeled target training data.



(a) Fr→En (b) En→Es

Figure 6: Sensitivity tests for hyper-parameters.

and feed it into a GRU for the representations of output, candidate and buffer. Among all these variants, *-hier* is most comparable, although worse than TAN. This shows the benefit of relation grouping in the transfer setting. Indeed, in the hierarchy of universal dependencies, higher level is more general to be shared across languages. By removing the auxiliary task (*-aux*), the performance drops considerably for the last four experiments. It is even worse when removing DAN (*-dan*). This shows either the auxiliary task or DAN alone is not good enough to capture shared information. By using auxiliary-conditional DAN, we can bridge the language gap through shared syntactic space. We also observe the effect of using partial dependency tree to compute the output representation, compared with a simple GRU (*-ptree*) or a complete dependency tree (*+ftree*). This indicates that RNN fails to capture syntactic structure and may produce unnecessary noise, and a complete dependency tree brings excessive knowledge about the words not contained in the stack, making it hard to propagate information when the path is long.

To show that our model indeed benefits from unlabeled target data during training and learns language-invariant features that are useful for predictions, we conduct experiments to vary the number of unlabeled target data for training. The percentage of the unlabeled target data increases from 0.0 to 1.0. We select two transfer settings: Es→Fr and En→Es and compare TAN with CL-DSCL and A-R^{2}NN. The inductive performances are shown in Figure 5. TAN shows more stable improvements with the increasing number of target training data compared with the other 2 baselines. This proves that TAN is able to learn from unlabeled target for adaptation.

We also conduct sensitivity tests on the hyper-parameters of TAN. As shown in Figure 6, we separately vary the trade-off parameter β for Fr→En and the hidden layer dimension of the configuration representation for En→Es. We plot both the transductive and inductive results on training and test data, respectively. The results are overall stable for β and change slowly when changing the dimension of hidden layer from 40 to 100. This proves the robustness and stableness of our proposed model against these variations.

6 Conclusion

We propose a novel transition-based deep model for cross-lingual aspect extraction that integrates domain adversarial network with syntactic information. Our model focuses on learning transferable configuration for the sentence at each timestamp and aims to learn a shared space that is action-sensitive but language-invariant given the syntactic structure.

Acknowledgements

We thank the support from NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020 and Fuji Xerox Corporation through joint research on Multilingual Semantic Analysis.

References

[Bérard *et al.*, 2016] Alexandre Bérard, Christophe Servan, Olivier Pietquin, and Laurent Besacier. MultiVec: a Multilingual and Multilevel Representation Learning Toolkit for NLP. In *LREC*, 2016.

[Chandar *et al.*, 2014] Sarath Chandar, Stanislas Lauly, Hugo Larochelle, Mitesh Khapra, Balaraman Ravindran, Vikas C Raykar, and Amrita Saha. An autoencoder approach to learning bilingual word representations. In *NIPS*. 2014.

[Chen *et al.*, 2016] Xilun Chen, Ben Athiwaratkun, Yu Sun, Kilian Q. Weinberger, and Claire Cardie. Adversarial deep averaging networks for cross-lingual sentiment classification. *CoRR*, 2016.

[de Marneffe *et al.*, 2014] Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. Universal stanford dependencies: A cross-linguistic typology. In *LREC*, 2014.

[Ding *et al.*, 2017] Ying Ding, Jianfei Yu, and Jing Jiang. Recurrent neural networks with auxiliary labels for cross-domain opinion target extraction. In *AAAI*, 2017.

[Dyer *et al.*, 2015] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-based dependency parsing with stack long short-term memory. In *ACL*, 2015.

[Ganin *et al.*, 2016] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1):2096–2030, 2016.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*. 2014.

[Gouws *et al.*, 2015] Stephan Gouws, Yoshua Bengio, and Greg Corrado. Bilbowa: Fast bilingual distributed representations without word alignments. In *ICML*, 2015.

[Hermann and Blunsom, 2014] Karl Moritz Hermann and Phil Blunsom. Multilingual models for compositional distributed semantics. In *ACL*, 2014.

[Hu and Liu, 2004] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *KDD*, 2004.

[Jakob and Gurevych, 2010] Niklas Jakob and Iryna Gurevych. Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *EMNLP*, 2010.

[Jin and Ho, 2009] Wei Jin and Hung Hay Ho. A novel lexicalized hmm-based learning framework for web opinion mining. In *ICML*, pages 465–472, 2009.

[Klementiev *et al.*, 2012] Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. Inducing crosslingual distributed representations of words. In *COLING*, 2012.

[Lample *et al.*, 2016] Guillaume Lample, Miguel Ballesteros, Kazuya Kawakami, Sandeep Subramanian, and Chris Dyer. Neural architectures for named entity recognition. In *Proc. NAACL-HLT*, 2016.

[Li *et al.*, 2012] Fangtao Li, Sinno Jialin Pan, Ou Jin, Qiang Yang, and Xiaoyan Zhu. Cross-domain co-extraction of sentiment and topic lexicons. In *ACL*, 2012.

[Liu *et al.*, 2015] Pengfei Liu, Shafiq Joty, and Helen Meng. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *EMNLP*, 2015.

[Meng *et al.*, 2012] Xinfan Meng, Furu Wei, Xiaohua Liu, Ming Zhou, Ge Xu, and Houfeng Wang. Cross-lingual mixture model for sentiment classification. In *ACL*, 2012.

[Odena *et al.*, 2017] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *ICML*, 2017.

[Prettenhofer and Stein, 2010] Peter Prettenhofer and Benno Stein. Cross-language text classification using structural correspondence learning. In *ACL*, 2010.

[Qiu *et al.*, 2011] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. Opinion word expansion and target extraction through double propagation. *Comput. Linguist.*, 37(1):9–27, 2011.

[Wan, 2009] Xiaojun Wan. Co-training for cross-lingual sentiment classification. In *ACL-AFNLP*, 2009.

[Wang *et al.*, 2016] Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. Recursive neural conditional random fields for aspect-based sentiment analysis. In *EMNLP*, 2016.

[Wang *et al.*, 2017] Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. Coupled multi-layer tensor network for co-extraction of aspect and opinion terms. In *AAAI*, 2017.

[Yin *et al.*, 2016] Yichun Yin, Furu Wei, Li Dong, Kaimeng Xu, Ming Zhang, and Ming Zhou. Unsupervised word and dependency path embeddings for aspect term extraction. In *IJCAI*, 2016.

[Zhou *et al.*, 2014] Joey Tianyi Zhou, Sinno Jialin Pan, Ivor W. Tsang, and Yan Yan. Hybrid heterogeneous transfer learning through deep learning. In *AAAI*, 2014.

[Zhou *et al.*, 2015] Guangyou Zhou, Tingting He, Jun Zhao, and Wensheng Wu. A subspace learning framework for cross-lingual sentiment classification with partial parallel data. In *IJCAI*, pages 1426–1432, 2015.

[Zhou *et al.*, 2016] Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. Cross-lingual sentiment classification with bilingual document representation learning. In *ACL*, 2016.