Multi-Domain Gated CNN for Review Helpfulness Prediction

Cen Chen
Ant Financial Services Group
Hangzhou, Zhejiang, China
chencen.cc@antfin.com

Minghui Qiu
Alibaba Group & Zhejiang University
Hangzhou, Zhejiang, China
minghui.qmh@alibaba-inc.com

Yinfei Yang
Mountain View, CA, United States
yangyin7@gmail.com

Jun Zhou
Ant Financial Services Group
Hangzhou, Zhejiang, China
jun.zhouju@antfin.com

Xiaolong Li
Ant Financial Services Group
Hangzhou, Zhejiang, China
xl.li@antfin.com

Jun Huang
Alibaba Group
Hangzhou, Zhejiang, China
huangjun.hj@alibaba-inc.com

Forrest Sheng Bao
Iowa State University
Ames, IA, United States
forrest.bao@gmail.com

ABSTRACT

Consumers today face too many reviews to read when shopping online. Presenting the most helpful reviews, instead of all, to them will greatly ease purchase decision making. Most of the existing studies on review helpfulness prediction focused on domains with rich labels, not suitable for domains with insufficient labels. In response, we explore a multi-domain approach that learns domain relationships to help the task by transferring knowledge from data-rich domains to data-deficient domains. To better model domain differences, our approach gates multi-granularity embeddings in a Neural Network (NN) based transfer learning framework to reflect the domain-variant importance of words. Extensive experiments empirically demonstrate that our model outperforms the state-of-the-art baselines and NN-based methods without gating on this task. Our approach facilitates more effective knowledge transfer between domains, especially when the target domain dataset is small. Meanwhile, the domain relationship and domain-specific embedding gating are insightful and interpretable.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Transfer learning**.

KEYWORDS

Review helpfulness prediction; transfer learning

ACM Reference Format:
Cen Chen, Minghui Qiu, Yinfei Yang, Jun Zhou, Xiaolong Li, Jun Huang, and Forrest Sheng Bao. 2019. Multi-Domain Gated CNN for Review Helpfulness Prediction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313587

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313587

1 INTRODUCTION

Product reviews, primarily text, provide important information for consumers to make purchase decisions. Hence, it makes great business sense to quantify the quality of reviews and present consumers useful reviews in an informative manner. In fact, major E-commerce websites like Amazon.com encourage users to rate whether a review is helpful or not. However, most reviews do not have helpfulness scores or votes. Fewer than 20% of the reviews in Amazon Review Dataset [20] have 5+ votes while only 0.44% have 100+ votes. In addition, the review voting itself may be biased [5, 8]. It is thus interesting yet important to automatically infer the helpfulness score of any review.

Recently, there are growing interests from both the academia and the industry on such *review helpfulness prediction task* [17, 19, 30, 31]. Pioneering work hypothesizes that helpfulness is an underlying property of the text, and uses handcrafted linguistic feature to study it [17, 19, 30, 31]. Unfortunately, most of them focus on domains with rich labels, and may not be practical for domains with insufficient data. For example, the "Electronics" domain from the Amazon Review Dataset has more than 354k labeled reviews, while "Watches" has less than 10k [30]. To alleviate, we study Transfer Learning (TL) for review helpfulness prediction. Specifically, we propose a multi-domain approach that transfers knowledge from multiple data-rich domains to data-deficient domains. Inspired by the remarkable performance and efficiency of Convolutional Neural Networks (CNNs) on many tasks in natural language processing [6, 13], we adopt CNNs to prove the concept of our TL framework for the task. Our proposed framework is a multi-domain gated NN model based on two important observations below (arranged in the order of the data processing flow.)

First, while embeddings of words are equally used downstream in previous applications of CNNs in NLP, actually different words have different and domain-dependent impacts to review helpfulness. For example, the top aspect extracted by the model in [30] for "Watches" domain is "Warranty" whereas "Portablity" for "Electronics" domain, implying that words related to "Warranty" are

important to "Watches," while words related to "Portability" are important to "Electronics." Hence, we introduce the *word-level gating* mechanism to weight embeddedings[1].

Although gating mechanisms have been used in RNNs [7, 26, 29], our gating method is complimentary for being word- and domain-specific, i.e., one gate weight for one word in one domain. Our gating mechanism can help selectively memorize the input representations of the words and scores the relevance/importance of such representations in different domains, to provide insightful word-level interpretations for the TL prediction results. Meanwhile, to the best of our knowledge, our gating mechanism is the first to study cross-domain word-level importance.

Second, domain difference is indeed a pending problem in review analysis, as agreed by previous work [6, 17, 30]. When transferring knowledge, not all domains are helpful to a target domain. We propose multi-domain relationship learning to exploit cross domain relationships to properly transfer knowledge from *related* domains with sufficient labeled data to domains with limited reviews. It is worth noting that, existing studies on this task only focus on a single product category or largely ignore the inner correlations between different domains [6]. Previous work show some features are domain-specific while others can be shared. For example, image quality features are only useful for cameras [30], while semantic features and argument-based features are applicable for all domains [17, 31]. While there are some common practices to transfer knowledge between domains [21, 32], such as using a shared NN. We urge *domain correlations* to be established before the knowledge can be transferred properly for review helpfulness prediction, otherwise, transferring the knowledge from a wrong source domain may backfire. We thus provide a holistic solution to both domain correlation learning and knowledge transfer by incorporating domain relationship learning in our framework.

Furthermore, we propose to enrich the word representation in a vanilla CNN by considering cross-domain *character-* and *aspect-level* representations. Character-level representations are notably beneficial for alleviating the out-of-vocabulary problem [2, 14], while aspect distribution provides another semantic view on words [30].

In all, our model explores cross-domain word representations, multi-domain word gating, and multi-domain relationship learning to help the task. As shown in Figure 1, our network has four types of layers: (1) shared embedding layers, (2) gating layers, (3) convolutional layers, and (4) output layers with relationship learning. Experimental results show that our final model can correctly tap into domain correlations and facilitate the knowledge transfer between correlated domains. In all, we summarize our contributions following the network architecture (left to right in Figure 1) below:

- **Rich word representations:** We study CNN for review helpfulness prediction task, and further enrich our word representations by incorporating multi-granularity information, i.e., character-, word-, and aspect- based embeddings;
- **Domain specific gating layer:** We proposed a novel domain-specific gating mechanism to learn/interpret the important/non-important words in reviews and to boost model performance.
- **Cross-domain relationship learning:** We are the first to study cross-domain relationship learning for the task and show that the cross-domain correlations learned are insightful and beneficial for the task;
- **Evaluation:** Extensive experiments show that our model significantly outperforms the state of the art and can transfer knowledge to help target domain with limited data.

[1]The gates are applied over all three types of representations (i.e., character-, word-, and aspect-based) for all words and weights are domain-specific.

2 RELATED WORK

Review helpfulness prediction is a task close to but *different from sentiment analysis* [16]. The goal of sentiment analysis is to detect the emotion behind an opinion, whereas review helpfulness aims to identify helpful information like reasoning etc., regardless of the emotion nor the opinion [17, 31].

Recent studies on review helpfulness prediction extract handcrafted features from the review text. For example, semantic features used in [31] and [19], aspect features in [30] and argument-based features in [17]. These methods require prior knowledge and human effort in feature engineering and may not be robust for new domains. CNNs with the ability to extract deep features, have demonstrated remarkable performance over many NLP tasks, for its high-efficiency and comparable performance to Recurrent Neural Networks (RNNs) [13]. Our initial study shows that vanilla RNN [27] does not outperform our CNN models. It echoes the finding that RNNs are more robust but do not guarantee to have better results than CNNs [33]. Thus, we employ CNNs for our task. Since character-level representations are notably beneficial for alleviating the out-of-vocabulary [14] problem, while aspect distribution provides another semantic view on words [30], we further enrich our word representation by adding character- and aspect-based representations.

Unlike a single domain model, our work focuses on leveraging reviews from multiple domains. As a word may play different importance on different domains, we consider to weight word representations by adding word-level gates. Note that some gating mechanisms have been used in RNNs [7, 10]. The work in [15] explored contextual word attentions in different tasks. Different from these studies, our word-level gates are domain-specific and help further differentiate important and non-important words in different domains.

It is common that some domains have rich user reviews while other domains may not. To help domains with limited data, we study cross-domain learning, i.e., transfer learning (TL) [22] or multi-task learning (MTL) [36] for this task. There are generally two lines of studies for TL. The first is supervised domain adaptation, which assumes enough labeled data from source domains and also a little labeled data from a target domain [9]. The second assumes having labeled data from source domains only and some unlabeled data from a target domain [3, 35]. Our study belongs to the first line of work. Typical method in this line is to find a shared feature space which can reduce the divergence between the distributions of the source and target domains [1, 28]. In this work, we adapt CNN model to learn shared feature representations.

With the popularity of deep learning, a great amount of Neural Network (NN) based methods are proposed for TL [6, 11, 12, 34]. A typical framework is to use a shared NN to learn shared features for

both source and target domains [21, 32]. Another approach is to use both a shared NN and domain-specific NNs to derive shared and domain-specific features [18]. A multi-domain relationship learning method is introduced in [37], to uncover the relationship between domains. Inspired by this, we adopt the relationship learning in our gated multi-domain framework to help model the correlation between different domains. Experiments show that the multi-domain relationship learning method indeed helps capture the correlations between different domains and can further boost our model's performance by better leveraging knowledge from correlated domains. We have also visualized the learned domain correlations and find them insightful.

3 MODEL

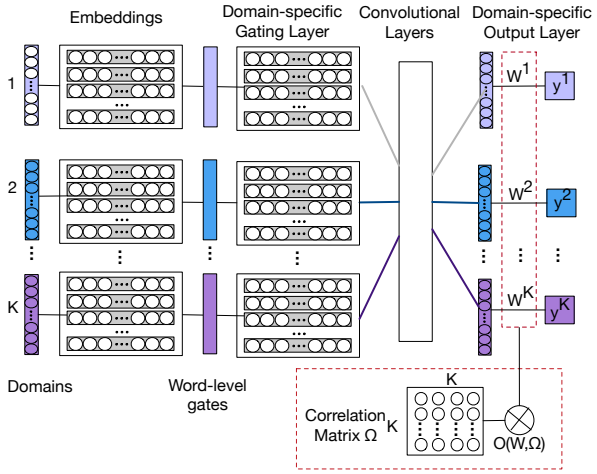

Figure 1: Proposed multi-domain gated CNN model. Note that the embedding and convolutional layers are shared across domains, while the gating and output layers are domain-specific. U and $W^{(\cdot)}$ are shared and domain-specific output layers, respectively. The parameter Ω models domain correlations.

Review helpfulness prediction is defined as a regression task to predict the helpfulness score given a text review [31]. The ground truth of helpfulness is determined using the "a of b approach": a of b users think the review is helpful.

In this work, we consider a *cross-domain review helpfulness prediction task* given labeled reviews from multiple source domains and a target domain. We seek to transfer knowledge from data-rich source domains to a target domain. For a review $\mathbf{X}^k$, our goal is to predict its helpfulness score y^k, where $k \in [0..K]$ is the domain label indicating which domain the data instance is from.

As shown in Fig. 1, Our proposed multi-domain gated CNN has four components: embedding layers, gating layers, convolutional layers, and output layers with relationship learning. Embedding and convolutional layers are shared across domains, while other layers are domain-specific.

3.1 Shared embedding layers

Our word embeddings contain word-level, character-level, and aspect-level representations.

3.1.1 Word and Character Representations. A review $\mathbf{X}$ consists of a sequence of words, i.e., $\mathbf{X} = [x_1, x_2, \ldots, x_m]$. Following the model in [13], for words in a review $\mathbf{X}$, we first lookup their embeddings $(\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m)$ from an embedding matrix $E \in \mathbb{R}^{|V| \times D}$ where $|V|$ is vocabulary size and D is embedding dimension, and $\mathbf{e}_i \in \mathbb{R}^{D \times 1}$. This word embedding matrix is then fed into a convolutional neural network to obtain an output representation. This is a typical *word embedding* based model.

In many applications, such as text classification [4] and machine reading comprehension [25], it is beneficial to enrich word embeddings with subword information. Inspired by that, we consider to use a character embedding layer to obtain *character embeddings* to enrich word representations. Specifically, the characters of the i-th word x_i are embedded into vectors and then fed into another convolutional neural network to obtain a fixed-sized vector $\texttt{CharEmb}(x_i)$.

3.1.2 Aspect Representations. A recent work in [30] shows that extracting the aspect/topic distribution from the raw textual contents does help the task of review helpfulness prediction. The reason is that many helpful reviews tend to mention certain aspects, like "brand," "functionality," or "price," of a product. Inspired by this, we enrich our word representations by aspect distributions. We adopt the generative model in [30] to learn aspect-word distributions $\Phi \in \mathbb{R}^{|V| \times A}$, where A is aspect size and $|V|$ is the size of vocabulary. Note, aspect representations learned are shared across domains.

A word-aspect representation Φ' is obtained by row-wise normalization of the matrix Φ. Then for each word x_i in input review $\mathbf{X}$, we obtain aspect representation by looking up the matrix Φ' to get $\Phi'_i \in \mathbb{R}^{A \times 1}$.

3.1.3 Putting all three embeddings together. In all, for an input review $\mathbf{X}$, we obtain its representation as:

$$\mathbf{e}_X = [\mathbf{e}'_1, \mathbf{e}'_2, \ldots, \mathbf{e}'_m], \tag{1}$$

$$\mathbf{e}'_i = \mathbf{e}_i \oplus \texttt{CharEmb}(x_i) \oplus \Phi'_i, \quad \forall i \in [1..m], \tag{2}$$

where $\mathbf{e}_i$, $\texttt{CharEmb}(x_i)$, and Φ'_i are word-, character-, and aspect-representations respectively, and $\oplus$ is a stacking operator. Note that m (=100 in this paper) is the sentence length limit. Sentences shorter than m words will be padded with zeros, while those longer than m words will be truncated.

3.2 Domain-specific gating layers

It is intuitive that given a domain, some words play more important roles to the helpfulness of reviews. For example, "warranty" words are more important for "Watches" domain than other domains, and descriptive or semantic words (such as "great battery life" or "versatile function") are more informative than general background words like "phone" in "Cellphone" domain [30]. Hence, we propose to weight the input word embeddings by a multi-domain gating mechanism. Specifically, the input word representations are first transformed with a weighting matrix $\mathbf{V}_g$ and concatenate with a domain-specific embedding $\mathbf{V}_k$ for domain k. The final output is then fed into a fully connected layer to obtain final word-level gates.

More specifically, for each input word x_i, we learn its weight g_i through the gating mechanism. And for input $\mathbf{X} = [x_1, \ldots, x_m]$, we obtain its representation $\mathbf{e'}_X$ as:

$$\mathbf{e}_X = [\mathbf{e}'_1, \mathbf{e}'_2, \ldots, e'_m], \tag{3}$$

$$g_i = \texttt{FC}(\mathbf{V}_g^\top \mathbf{e}'_i \oplus \mathbf{V}_k), \quad \forall i \in [1..m], \tag{4}$$

$$\mathbf{e'}_X = [g_1\mathbf{e}'_1, g_2\mathbf{e}'_2, \ldots, g_m\mathbf{e}'_m], \tag{5}$$

where $\mathbf{V}_g$ is a transform matrix, $\mathbf{V}_k$ is domain-specific weights, and FC represents fully connected layers with sigmoid activations. The same weight g_i is applied to all dimensions of the embedding $\mathbf{e}'_i$ for word x_i.

3.3 Shared convolutional layers

Next, we stack a 2-D convolutional layer and a 2-D max-pooling layer on the matrix $\mathbf{e}'_X$ to obtain the hidden representation h_X. Multiple filters are used, where for each, we obtain a hidden representation h_f. All these representations are concatenated to form the final representation h_X.

$$h_f = \texttt{CNN}\Big(\mathbf{e}'_X, \text{filterSize} = \big(f, D, C\big)\Big),$$
$$h_X = h_2 \oplus h_3 \oplus h_4 \oplus h_5. \tag{6}$$

where $f \in \{2, 3, 4, 5\}$ is window size, D is embedding dimension, C is channel size, and $\texttt{CNN}(\cdot)$ represents a convolution layer followed by a max-pooling layer [2]. We refer our base model as Embedding-Gated CNN (EG-CNN).

3.4 Domain-specific output layers

If all the domains are homogeneous, we can build an unified model for our task to optimize the following objective:

$$l = \sum_k \sum_{\mathbf{X}^k \in \mathcal{D}^k} \left(\mathbf{U}^\top \texttt{EG-CNN}\left(\mathbf{X}^k\right) - y^k\right)^2 + l_{reg}, \tag{7}$$

where $\mathbf{U}$ is the output layer weights, $\mathbf{X}^k$ is a review in the set $\mathcal{D}^k$ containing all reviews from domain k, y^k is the corresponding label, l_{reg} is a regularization term.

However, *domains are not homogeneous.* The formulation in Eqn. (7) is limited because it does not take domain differences into consideration. To utilize cross-domain knowledge, we convert the method above to a multi-domain setting where besides the shared output layer $\mathbf{U} \in \mathbb{R}^{P\times 1}$, we add a domain-specific output layer $\mathbf{W}_k$ for each domain k. Assuming $\mathbf{W}_k \in \mathbb{R}^{P\times 1}$, where P is output layer size, we have $\mathbf{W} = [\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_K]$. Unlike a unified model to learn universal feature representation, our *cross-domain relationship learning* approach has two output layers $\mathbf{U}$ and $\mathbf{W}$ to model domain *commonalities* and *differences* respectively.

Following matrix-variate distribution setting in [37], we assume:

$$\mathbf{W} = q(W) \prod_{i=1}^{K} \mathcal{N}\left(W_i \mid \mathbf{0}_P, \epsilon_i^2 \mathbf{I}_P\right), \tag{8}$$

$$q(W) = \mathcal{MN}_{P\times K}\Big(W \mid \mathbf{0}_{P\times K}, \mathbf{I}_P \times \Omega\Big), \tag{9}$$

where $\mathcal{N}(\cdot \mid A, B)$ and $\mathcal{MN}(\cdot \mid A, B)$ are normal distribution and matrix-variate normal distribution with A as mean and B as variance/covariance matrix respectively, $\mathbf{0}$. is a zero vector/matrix, $\mathbf{I}_P$ is identity matrix. The covariance matrix $\Omega \in \mathbb{R}^{K\times K}$ models domain correlations, where $\Omega_{i,j}$ is the correlation between domains i and j.

As shown in [37], the above formulas are resolved to optimize the trace of the matrix product $\text{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^\top)$. Hence, our objective can be defined as follows:

$$\begin{aligned} l \quad = \quad & \sum_k \sum_{X^k \in \mathcal{D}^k} ((\mathbf{U} + \mathbf{W_k})^\top \texttt{EG-CNN}(\mathbf{X}^k) - y^k)^2 + \\ & \lambda_1 \text{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^\top) + \lambda_2 l_{reg}, \\ \text{s.t.} \quad & \Omega \succeq 0, \quad \text{tr}(\Omega) = 1. \end{aligned} \tag{10}$$

where $\text{tr}(\cdot)$ gets the trace of a matrix, l_{reg} is a regularization term, λ_1 and λ_2 are weight coefficients. To minimize the trace term $\text{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^\top)$, when domain i and domain j are close, i.e., $\mathbf{W}_i$ is close to $\mathbf{W}_j$, the model tends to learn a large $\Omega_{i,j}$. And when domain i and j are distant, the model tends to learn a small $\Omega_{i,j}$.

3.5 The final model

In all, our final model has four types of layers: embedding layers, gating layers, convolutional layers, and output layers. In the output layers, we consider learning domain correlations in the multi-domain setting. Note that, if we set the correlation matrix Ω as an identity matrix (no domain correlation) and $\mathbf{U} = \mathbf{0}$ (no shared output layer), the multi-domain setting is degenerated to a fully-shared setting in [21]. The limitation of the fully-shared setting is that it ignores domain relationships. However, in practice, we may think that reviews from some domains (e.g., the domain "Electronics") are helpful to the task on other domains (e.g., "home" and "cellphones") but not all. With our model, we seek to automatically capture such domain relationships and use that information to help boost model performance.

4 INFERENCE

Let $\boldsymbol{\theta}$ be the model parameters in our model. It is not easy to optimize the objective function in Eqn. (10) with respect to Ω, $\mathbf{W}$, $\mathbf{U}$, and $\boldsymbol{\theta}$. Alternatively, we use a stochastic alternating method [37] to first optimize $\boldsymbol{\theta}$, $\mathbf{W}$, and $\mathbf{U}$ through fixing Ω, and then optimize Ω by fixing $\boldsymbol{\theta}$, $\mathbf{W}$, and $\mathbf{U}$. The procedure is repeated within a given iteration or until the convergence criterion is met.

Optimizing $\mathbf{W}$, $\mathbf{U}$ *and* $\boldsymbol{\theta}$. While Ω is fixed, the objective function is convex. We can use gradient descent method to optimize $\mathbf{W}$, $\mathbf{U}$, and $\boldsymbol{\theta}$ jointly. The gradients can be computed by back propagation method with details in [24].

Optimizing Ω. With the fixed $\mathbf{W}$, $\mathbf{U}$, and $\boldsymbol{\theta}$, our objective is to optimize $\mathbf{min}_\Omega\ \mathbf{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^\top)$ with regard to $\Omega \succeq \mathbf{0}$ and $\mathbf{tr}(\Omega) = \mathbf{1}$.

According to [37], the convex problem above has a closed-form solution: $\Omega = \frac{(\mathbf{W}^\top\mathbf{W})^{\frac{1}{2}}}{\text{tr}\left((\mathbf{W}^\top\mathbf{W})^{\frac{1}{2}}\right)}$. If $\mathbf{W_i}$ and $\mathbf{W_j}$ are close, $\Omega_{\mathbf{i,j}}$ will be be large. $\mathbf{W}^\top\mathbf{W}$ is a symmetric matrix, whose square root can be computed by singular-value decomposition (SVD).

[2] Our gated TL framework can work with any sentence representation model, where CNN layer here is chosen as an example to prove the concept for the task.

5 EXPERIMENTS

We use reviews from 5 different categories/domains from the public Amazon Review Dataset [20] in the experiments. The 5 categories are: watches, Phones, Outdoor, Home, and Electronics, each with 9k, 18k, 73k, 219k, 354k data instances respectively. The 5 categories are chosen in accordance with the practices of previous work on this topic [6, 30, 31].

In our model, we initialize the lookup table $\mathbf{E}$ with pre-train word embeddings from *GloVe* [23] by setting $\boldsymbol{D = 100}$. For aspect representation, we adopt the settings from [30] to set the aspect size to 100. For the convolutional layer, the activation function is ReLU, the channel size is set to 128. The output layer size of $\mathbf{U}$ and $\mathbf{W}_{\boldsymbol{k}}$ are 128, i.e., $\boldsymbol{P = 128}$. We use data from five domains, hence $\boldsymbol{K = 5}$. AdaGrad is used with an initial learning rate of 0.08.

Following the previous work on helpfulness prediction, all experiment results are evaluated in the *Pearson correlation coefficient* between the predicted helpfulness score and the ground truth score. The ground truth scores are computed by "a of b approach", indicating the percentage of consumers thinking a review as useful.

5.1 EG-CNN vs. the state of the art and vanilla CNNs without gating

We first compare EG-CNN, i.e., our full base model with rich word representations and domain-specific gates, against those based on traditional linguistic features. Following [31], five baselines use handcrafted features "STR", "UGR", "LIWC", "INQ" and aspect-based features "ASP" [30]", respectively, are established. We further consider two fusional baselines with ensemble linguistic features: Fusion_1 with "STR", "UGR", "LIWC", and "INQ" features [31] and Fusion_2 with additional "ASP" features [30]. Different vanilla CNN-based models without gating are also compared to examine the effectiveness of different embeddings and domain-specific gates: CNN - the vanilla CNN model in [13], $\text{CNN}_{\boldsymbol{c}}$ - the vanilla CNN model with additional character-based representation used in [6].

Table 1: Performance of EG-CNN variants and linguistic feature baselines in Pearson correlation coefficient. † and ‡ mean the method outperforms the best baseline with $\boldsymbol{p < 0.05}$ and $\boldsymbol{p < 0.1}$ by the Student's paired t-test in terms of prediction errors respectively.

		Watches	Phones	Outdoor	Home	Elec.
Baselines	STR	0.276	0.349	0.277	0.222	0.338
	UGR	0.425	0.466	0.412	0.309	0.355
	LIWC	0.378	0.464	0.382	0.331	0.400
	INQ.	0.403	0.506	0.419	0.366	0.405
	ASP	0.406	0.437	0.385	0.283	0.406
	Fusion_1	0.488	0.539	0.497	0.432	0.484
	Fusion_2	0.493	0.550	0.501	0.436	0.491
CNNs without gating	CNN	0.480	0.562	0.501	0.459	0.524
	$\text{CNN}_{\boldsymbol{c}}$	0.495	0.566	0.511	0.464	0.521
Our method	EG-CNN	$\mathbf{0.515}^{\dagger}$	$\mathbf{0.585}^{\dagger}$	$\mathbf{0.555}^{\dagger}$	$\mathbf{0.541}^{\dagger}$	$\mathbf{0.544}^{\ddagger}$

Table 1 shows several interesting observations that validate our motives behind this work. First, even the vanilla CNN outperforms the baselines, indicating the expressive superiority of NN-extracted features over handcrafted features. Second, increasing the level of embedding yields better performance as vanilla CNN is outperformed by $\text{CNN}_{\boldsymbol{c}}$. In all cases, EG-CNN outperforms all the CNN variants, which shows the importance of adding aspect-based representations and the gating mechanism of our model. Last, EG-CNN consistently outperforms all the baselines. This further shows the advantage of our base model, i.e., EG-CNN, for the task.

5.2 Our full model with domain relationships

To evaluate the effectiveness of our domain relationship learning, we compare our proposed full model against the following baselines:

- Source-only (Src-only): EG-CNN model that does not consider target data. We use the largest domain 'Elec.' as source.
- Target-only (Tgt-only): EG-CNN model that does not consider source data, it is equivalent to train our model on each domain individually by setting $\mathbf{W = 0}$.
- Fully-Shared transfer learning model (FS): a variant of our full model by setting $\mathbf{U = 0}$, which is identical to the fully-shared TL model in [21] except that our base model is EG-CNN with rich word representations;
- Transfer Learning with Adversarial loss (TL_{adv}): a recently proposed TL model for the task with adversarial loss [6], which is extended from [18]. For fair comparison, the base model used in TL_{adv} is also replaced with EG-CNN.
- Our full model (Ours): our final model with both $\mathbf{U}$ and $\mathbf{W}$ in a domain relationship learning setting.

Table 2: Comparisons between our model and TL models

		Watches	Phones	Outdoor	Home	Elec.
Baselines	Src-only	0.481	0.502	0.501	0.445	0.544
	Tgt-only	0.515	0.585	0.555	0.541	0.544
	FS	0.522	0.580	0.551	0.518	0.534
	TL_{adv}	0.525	0.588	0.556	0.541	0.545
	Ours	$\mathbf{0.535}^{\dagger}$	$\mathbf{0.592}^{\ddagger}$	$\mathbf{0.561}^{\dagger}$	**0.544**	**0.548**

According to results in Table 2, Tgt-only outperforms Src-only, which is intuitive as source domains are close to but different from the target domain. On all product domains, our model consistently achieves better results than Src-only, Tgt-only, FS, and $\text{TL}_{\boldsymbol{adv}}$, supporting the effectiveness and benefit of multi-domain relationship learning. The improvement is greater on domains with fewer labeled data, e.g., the "Watches" domain which has the least number of reviews and where our model shows the most improvement.

Interestingly, the FS model performs much worse than the Tgt-only model in the "Home" domain. This might be justified by the potential domain shift, under which the FS model may not perform better than the Tgt-only model. Because some domains are more related while some others are more different, incorporating data from those less related can hardly help, especially when the target domain (such as "Home") has sufficient data for the Tgt-only model to perform well enough.

The domain correlations presented in Figure 2 provides further evidence into this. The "Home" domain is positively correlated with the two smallest domains, "Watches" and "Phones," and negatively correlated with two other data-rich domains, "Outdoor" and "Electronics." When relevant domains have much less data than

uncorrelated domains, the FS model will cause non-ideal transfer of knowledge, just like what we have seen on the "Home" domain. On a separate note, our model is more robust that it outperforms all the baselines, thanks to our model's ability to learn domain correlations and leverage knowledge from the right domains.

5.3 Is more data equal to more knowledge?

We study whether more data equals to more knowledge by varying the amount of source or target data available. A transfer learning setting from the largest domain "Electronics" to a target domain "Phones" is used as a case study.

Table 3: Performances w.r.t. different amounts of target data.

% of Phone Data	25%	50%	75%	100%
# of Reviews	19.7k	39.4k	59.2k	78.9k
Target-only	0.453	0.532	0.564	0.585
Our full model	0.481	0.544	0.575	0.594
Improvement	6.2%	2.3%	2.0%	1.5%

We study how performance changes along with different amounts (1/4, 1/2, 3/4, and 100% of all, data pre-scrambled) of reviews from the target domain used in the case study. The performances of both our full model and the Tgt-only model are presented in Table 3. Our model has the most advantage (6.2%) over Tgt-only model when only 25% of target domain data is used. The more target data involved in training, the smaller the advantage is. When all of the target domain data is used, the advantage shrinks to 1.5%. It suggests that our model helps more when the target domain is more data-deficient. The reason behind is that the task on target domain benefits little from the knowledge transferred from other domain when target domain data itself is enough to train a good model.

5.4 Domain relationships

We lastly examine the *domain correlation* Ω learned by our model. The correlation matrix is a 5×5 matrix where each value indicating the correlation between two domains. As in Figure 2, domain correlation learned is reasonable. For example, domain "Phone" is close to "Home" and "Elec.," but not so close to "Watch" and "Outdoor."

	Watch	Phone	Outdoor	Home	Elec.
Watch	1	-0.096	-0.41	0.128	-0.008
Phone	-0.096	1	-0.089	0.166	0.206
Outdoor	-0.41	-0.089	1	-0.2	-0.014
Home	0.128	0.166	-0.2	1	-0.279
Elec.	-0.008	0.206	-0.014	-0.279	1

Figure 2: An illustration on domain correlation. A redder cell represents a closer domain relationship between the two domains.

5.5 Interpretability of embedding gates

To examine the word-level gates in our proposed model, we visualize the gate weights learned in Figure 3. We randomly choose one sentence from the review text from the 'Watch' domain and plot a gray-scale heat map according to the gate weights learned for the words. The weights, ranging from 0 (unimportant) to 1 (important), are placed under each word. The redder a cell is, the closer a word's weight is to 1.

i	be	really	disappoint	in	this	product	i
0.2517	0.2405	0.4392	0.5086	0.4541	0.2902	0.3993	0.2517
have	a	watch	that	i	could	not	get
0.4271	0.3001	0.3395	0.336	0.2517	0.4097	0.3348	0.4114
the	pin	all	the	way	back	in	purchase
0.3385	0.3949	0.3814	0.3385	0.4131	0.4975	0.4541	0.4106
this	hope	to	resolve	the	problem	the	pin
0.2902	0.5286	0.3529	0.6678	0.3385	0.5587	0.3385	0.3949
bit	bent	and	could	not	handle	the	task
0.5564	0.5617	0.3673	0.4097	0.3348	0.5445	0.3385	0.5777

Figure 3: An visualization of word-level gate weights, ranging from 0 (unimportant) to 1 (important) from a review sample in the 'Watch' domain. The redder a cell is, the closer a word's weight is to 1.

The example in Figure 3 is very interpretable. Stop words, e.g., "i", "be"[3], and "the", and general background words, such as "product" and "watch" in the "Watches" domain, have small weights, whereas sentiment words and adjectives are assigned with large weights, such as "disappoint," "careful," and "error." In summary, we find that the word-level gate weights learned are helpful for improving model performance and provide insightful word-level interpretations for the prediction.

Furthermore, we visualize the top important and non-important words indicated by the word-level gates learned by our model in all the five domains in Table 4. First, important words and non-important words are easily distinguishable. For example in the "Watch" domain, important words include words related to opinions like "angry" and "disappointment," or aspects like "warranty." And non-important words contain stop words like "i," "be," and some words that do not carry specific meanings like "le" and "de." Similarly in other domains, we find the top important words are far more meaningful than the unimportant words.

Table 4: Top important/unimportant words in five domains.

Domain	Top Important Words
Watch	cut-out,flimsier,unsuccesfully,angry,warranty,disappointment
Phone	unplayable,retransmit,equipped,trouble-free,screen,well-protected
Outdoor	bizzare,wired,niggles,slumberland,stuggle,double-a,right-angle
Home	unsuccesfully,bizzare,button-down,stone-age,creasing,wierd,home
Elec.	units,superiores,unsuccesfully,know-nothing,exterior,checkable
Domain	**Top Unimportant Words**
Watch	le,de,como,i,bonito,be,monte,hoy,ser,in,the,alta,-,um,mil,monte
Phone	para,le,por,la,eso,todo,mudo,como,hace,gran,the,-,musica,casa
Outdoor	ha,hoy,de,le,an,como,the,um,despues,ecuador,6,-,vinal,mas,this,km
Home	mi,por,ser,ha,lo,como,eso,nah,the,quart,doubly,para,alta,estas,fue
Elec.	high,optional,fiber,play,an,this,meter,is,playing,inch,-,mb,the,ha

Second, in general, each domain has its own set of important words. For example,"warranty" in "Watches," and "trouble-free" and "screen" in "Phones." And the top 10 important words in "Watches" domain differ from those in "Phones." Interestingly, some words are important across different domains. For example, "unsuccessfully"

[3] All the words in our model are lemmatized.

is important to "Watches," "Home,", and "Electronics" domains, and "bizzare" is important to "Outdoor" and "Home" domains.

Last but not least, almost all of the top 10 unimportant words are hard to make sense of, e.g. words like "de," "le," and "lo." We find some background words like "fiber," "meter," and "play" are regarded as unimportant for "Electronics". This is also intuitive as the high occurrences of these words in "Electronics" domain makes them less important.

To summarize, the multi-domain word-level gates are interpretable and provide us an insight into the important and unimportant words in different domains. And such word-level gates can help boost model performance.

6 CONCLUSION

In this work, we tackle the review helpfulness prediction in a cross domain transfer learning framework. We built our base model on EG-CNN with word-, character- and aspect-based representations. On top of this model, domain-specific gates and domain relationships were learned to better transfer knowledge across domains. Extensive experiments showed that our model significantly outperforms the state of the art, and the domain relationships and embedding gates learned are insightful.

REFERENCES

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2007. Multi-task feature learning. In *NIPS*. 41–48.

[2] Miguel Ballesteros, Chris Dyer, and Noah A Smith. 2015. Improved transition-based parsing by modeling characters instead of words with LSTMs. *arXiv:1508.00657* (2015).

[3] John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP*. 120–128.

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *CoRR* abs/1607.04606 (2016). http://arxiv.org/abs/1607.04606

[5] Qing Cao, Wenjing Duan, and Qiwei Gan. 2011. Exploring determinants of voting for the "helpfulness" of online user reviews: A text mining approach. *Decision Support Systems* 50, 2 (2011).

[6] Cen Chen, Yinfei Yang, Jun Zhou, Forrest Sheng Bao, and Xiaolong Li. 2018. Cross-Domain Review Helpfulness Prediction Based on Convolutional Neural Networks with Auxiliary Domain Discriminators. In *NAACL*.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555* (2014).

[8] Cristian Danescu-Niculescu-Mizil, Gueorgi Kossinets, Jon Kleinberg, and Lillian Lee. 2009. How Opinions Are Received by Online Communities: A Case Study on Amazon.Com Helpfulness Votes. In *WWW*. 141–150.

[9] Hal Daume III. 2007. Frustratingly Easy Domain Adaptation. In *ACL*. 256–263.

[10] Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. 2016. Gated-attention readers for text comprehension. *arXiv:1606.01549* (2016).

[11] Xin Huang, Yuxin Peng, and Mingkuan Yuan. 2017. Cross-modal Common Representation Learning by Hybrid Transfer Network. *CoRR* abs/1706.00153 (2017).

[12] Xin Huang, Yuxin Peng, and Mingkuan Yuan. 2017. MHTN: Modal-adversarial Hybrid Transfer Network for Cross-modal Retrieval. *CoRR* abs/1708.04308 (2017).

[13] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *EMNLP*, 1746–1751.

[14] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-Aware Neural Language Models.. In *AAAI*. 41–49.

[15] Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. 2018. Hierarchical Attention Transfer Network for Cross-Domain Sentiment Classification. In *AAAI*.

[16] Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies* 5, 1 (2012), 1–167.

[17] Haijing Liu, Yang Gao, Pin Lv, Mengxue Li, Shiqiang Geng, Minglan Li, and Hao Wang. 2017. Using Argument-based Features to Predict and Analyse Review Helpfulness. In *EMNLP*. Copenhagen, Denmark, 1358–1363.

[18] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-task Learning for Text Classification. In *ACL*. 1–10.

[19] Lionel Martin and Pearl Pu. 2014. Prediction of Helpful Reviews Using Emotions Extraction. In *AAAI*. 1551–1557.

[20] J McAuley and J Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*. 165–172.

[21] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How Transferable are Neural Networks in NLP Applications?. In *EMNLP*. 479–489.

[22] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[23] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*.

[24] D.E. Rumelhart, G.E. Hintont, and R.J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323 (1986).

[25] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *CoRR* abs/1611.01603 (2016). http://arxiv.org/abs/1611.01603

[26] Chuanqi Tan, Furu Wei, Qingyu Zhou, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. 2018. Context-Aware Answer Sentence Selection With Hierarchical Gated Recurrent Neural Networks. *TASLP* 26, 3 (2018), 540–549.

[27] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*. 1422–1432.

[28] Chang Wang and Sridhar Mahadevan. 2008. Manifold alignment using procrustes analysis. In *ICML*. 1120–1127.

[29] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated Self-Matching Networks for Reading Comprehension and Question Answering. In *ACL*. 189–198.

[30] Yinfei Yang, Cen Chen, and Forrest Sheng Bao. 2016. Aspect-Based Helpfulness Prediction for Online Product Reviews. In *ICTAI*.

[31] Yinfei Yang, Yaowei Yan, Minghui Qiu, and Forrest Bao. 2015. Semantic Analysis and Helpfulness Prediction of Text for Online Product Reviews. In *ACL-IJCNLP*. 38–44.

[32] Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *ICLR*.

[33] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. 2017. Comparative Study of CNN and RNN for Natural Language Processing. *CoRR* abs/1702.01923 (2017). arXiv:1702.01923 http://arxiv.org/abs/1702.01923

[34] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *NIPS*. 3320–3328.

[35] Jianfei Yu and Jing Jiang. 2016. Learning Sentence Embeddings with Auxiliary Tasks for Cross-Domain Sentiment Classification. In *EMNLP*. 236–246.

[36] Yu Zhang and Qiang Yang. 2017. A survey on multi-task learning. *arXiv:1707.08114* (2017).

[37] Yu Zhang and Dit-Yan Yeung. 2010. A convex formulation for learning task relationships in multi-task learning. In *UAI*.