Adaptive Training Instance Selection for Cross-Domain Emotion Identification

Wenbo Wang
GoDaddy Inc.
Sunnyvale, California 94089
wwang@godaddy.com

Lu Chen
LinkedIn Corporation
Sunnyvale, California 94085
lchen2@linkedin.com

Keke Chen
DIAC Lab, Kno.e.sis Center, Wright State University, Dayton, 45435
keke.chen@wright.edu

Krishnaprasad Thirunarayan
Kno.e.sis Center, Wright State University, Dayton, 45435
t.k.prasad@wright.edu

Amit P. Sheth
Kno.e.sis Center, Wright State University, Dayton, 45435
amit@knoesis.org

ABSTRACT

This paper exploits a large number of self-labeled emotion tweets as the training data from the source domain to improve emotion identification in target domains (i.e., blogs and fairy tales), where there is a short supply of labeled data. Due to the noisy and ambiguous nature of self-labeled emotion training data, the existing domain adaptation methods that typically depend on high-quality labeled source-domain data do not work satisfactorily. This paper describes an adaptive source-domain training instance selection method to address the problem of noisy source-domain training data. The proposed approach can effectively identify the most informative training examples based on three carefully designed measures: consistency, diversity, and similarity. It uses an iterative method that consists of the following steps in each iteration: selecting informative samples from the source domain with the informativeness measures, merging with the target-domain training data, evaluating the performance of learned classifier for the target domain, and updating the informativeness measures for the next iteration. It stops until no new training instance is selected or in a designated number of iterations. Experiments show that our approach performs effectively for cross-domain emotion identification and consistently outperforms baseline approaches across four domains.

CCS CONCEPTS

• **Information systems** → **Sentiment analysis**; • **Computing methodologies** → *Natural language processing*;

KEYWORDS

Cross-Domain Emotion Identification, Instance selection

WI '17, Leipzig, Germany

978-1-4503-4951-2/17/08…$15.00
DOI: 10.1145/3106426.3106457

1 INTRODUCTION

Emotion identification aims to automatically identify people's fine-grained emotions (e.g., anger, disgust, fear, joy, sadness, and surprise) expressed in text. As the emotion-rich content grows rapidly on the Web, there is an increasing need to develop tools and techniques for emotion identification from various domains. Some research efforts have been devoted to identifying emotions from fairy tales and blog posts [1, 2].

A primary bottleneck to date for emotion identification is the lack of sufficiently large labeled training data. Statistical classification algorithms usually require a large amount of labeled data to train a reliable classifier. However, manually labeling emotions in text is labor-intensive and time-consuming. Moreover, compared with other annotation tasks such as entity or topic detection, a human annotator's judgment of emotions in text might be different from the ones an author intended to convey, and hence, both the size and the quality of training data are the bottlenecks for successful emotion identification.

Domain adaptation [14, 27] has emerged as an effective approach to addressing the problem of insufficient high-quality training data. It assumes the same classification problem in different domains share some inherent properties and thus one domain's training data can help another domain's classifier training. Typically, the source domain has a large labeled training dataset, while the target domain has very few labeled training instances. In emotion identification, it has been challenging to apply domain adaptation as there is no established high-quality labeled source-domain training data.

To tackle the challenge of large labeled training data, a recent study [29] exploits emotion hashtags in tweets to automatically infer their emotion labels. For example, we can obtain the following instance ⟨"Exactly one month until christmas! Woot **#excited**", *joy*⟩, where the trailing emotion hashtag "#excited" is stripped from the tweet and is used to label this tweet with emotion *joy*. We call this label extracting process as "self-labeling". In this way, a large number of "self-labeled" emotion tweets can be automatically collected from Twitter. It is appealing to adapt these tweets to help identify emotions in target domains (e.g., blogs and fairy tales) where the labeled instances are in short supply.

However, the extracted labels might be noisy or ambiguous. We show a few noisy tweets as an example: 1) the emotion label of a tweet might not be consistent with its content. A tweet may convey a mixture of emotions, whereas not all the emotions can be

Table 1: Emotion tweets: the emotion label in front of each tweet is inferred from the emotion hashtag in bold; informal expressions (misspellings, abbreviations, etc.) are under-waved.

#1	Fear: "Amazing night with my baby. Hope she liked our anniversary present. Alil early but whatever. :) hopefully tmmrw goes as planned. **#fear**"
#2	Sadness: "Why does my phone have to die so early in the morning. #canttweet **#depressing"**
#3	Anger: "My phone batt dies so quiick ....! **#annoyed"**

inferred if the author did not include hashtags for all the embedded emotions. The first half of tweet #1 in Table 1 conveys emotion *joy* that is not included in its label – *fear*; 2) features of Twitter and target domains (e.g., blogs and fairy tales) are different. As Table 1 shows, informal expressions, such as misspellings ("quiick" in tweet #3), abbreviations ("Alil" and "tmmrw" in tweet #1, "batt" in tweet #3) and multi-word concatenations ("#canttweet" in tweet #2) are common on Twitter. However, these expressions are rarely used in target domains.

Existing domain adaptation methods often consider the source-domain training data has high-quality labels and take all training data in domain adaptation. For example, Jiang et al. [16] merge the entire source-domain data with the target-domain training data, while overweighting the target-domain data in training. However, we argue that it is more beneficial to include the informative source-domain data than to include all of the source-domain data, which will be confirmed in the experiment section. Moreover, the majority of the domain adaptation methods are limited to binary sentiment classification but our problem involves multiple emotion classes. For example, Dai et al. [8] propose a variation of boosting to weight training instances, which handles binary classification only.

One may also wonder that the labeled training data in sentiment analysis [5, 26, 28] can be possibly applied to emotion identification. Sentiment analysis has been studied for more than a decade and accumulated sizable labeled examples that seem to share some labels with emotion analysis. However, there are several factors making it difficult to use such training data. First, most studies in sentiment analysis deal with binary classification while emotion identification is a multi-class fine-grained classification problem. The labels between the two domains, although sharing some similarity, are not consistent. Second, the majority of research on sentiment analysis deals with richer information, e.g., a product review document, while we focus on identifying emotions from more limited context, e.g., a sentence. This difference will very likely result in different feature distributions. Therefore, training data for sentiment analysis cannot be effectively used in emotion identification.

To address the above challenges, we propose an adaptive framework to iteratively select informative tweets out of self-labeled noisy tweets to enrich the target domain training data. An emotion identification classifier will be trained on the enriched training data to achieve a better accuracy for target domain emotion identification. Our framework has three unique contributions. (1) The source-domain instances are selected based on the three carefully designed *informativeness* measures: consistency, diversity, and similarity. *Consistency* measures the confidence of a tweet's label being consistent with its content, estimated by the labeled data from both source and target domains. *Diversity* encourages the selection of source instances containing features that are infrequent or underrepresented in the target domain. *Similarity* promotes source instances that are very similar to target domain instances. (2) The process can adaptively and progressively extract source-domain instances with the dynamically updated informativeness measures in each iteration. Its adaptive nature makes the process converge quickly with the outcome classifiers outperforming those by other approaches. (3) We have done an extensive evaluation on four target domains. Results show that our approach is effective for cross-domain emotion identification and consistently outperforms several baseline approaches.

2 RELATED WORK

Recently emotion-related studies are getting increasing attentions: leveraging emotions to predict review helpfulness [21], extracting emotion lexicons [6], intervening emotion contagion in social network [17], and predicting players' emotional responses during games [25]. When it comes to text-based emotion identification, many rule-based approaches [23, 24, 36] have been proposed. These approaches usually derive the emotion strength score of a sentence based on its component words whose emotion strength scores are predefined in a knowledge base. Several efforts on supervised methods [13, 30, 31] have been tested on a few thousands of sentences, partly due to the labor intensive nature of the manual labeling task. To ease the labeling task, there are some studies on how to automatically create labeled emotion dataset at the document [18], sentence [33], and tweet [22, 29, 32] levels.

Domain adaptation has attracted attention recently [27]. Previous work along different lines includes techniques for domain adaptation via regression-tree adaptation [7], feature alignment [26], dimensionality reduction [28], deep learning [14], and boosting [34]. Our work differs from prior studies in the following ways. First, it can identify fine-grained multi-class emotions (e.g., anger, disgust, fear, joy, sadness, and surprise) from text. Despite the large body of prior studies on domain adaptation, most of them deal with binary sentiment classification only [5, 14, 26, 34]. Second, the problem of identifying emotions from sentences is more challenging than that of binary polarity classification on product reviews [5, 14, 26, 28], because the context information of the former (i.e., sentence) is more limited than that of the latter (i.e., review document). Third, to reduce labeling efforts, we use self-labeled noisy emotion tweets instead of manually-labeled high quality ones as source domain data. Our adaptive framework can select informative tweets out of the noisy tweets and skip the low quality ones.

Since we explore this problem from the perspective of selecting informative instances, and hence limit our attention to the instance-based approaches. Jiang and Zhai [16] train an adaptive classifier on the union of both source and target domain instances, where the target domain labeled data are assigned larger weights since they are more representative of the target domain. Dai et al. [8] extend the AdaBoost algorithm to adjust the weights of training instances. Some studies [16, 37] apply a classifier trained on target

domain labeled data to identify "good" and "bad" instances from source data: a source instance is considered to be a good (or bad) one if it can be correctly (or incorrectly) classified by the classifier. In contrast, we attempt to select *informative* instances from the incorrectly classified source domain instances rather than the correctly classified source domain ones, because the fact that the classifier incorrectly classified some source domain instances may suggest that those instances contain information that the target domain training data lacks. Data selection has been frequently used in machine translation to select sentences that are very similar to those in target data [3, 12, 15, 19]. By doing so, the formed training data can hopefully better match the target data in text contents. However, we find that selecting tweets that are similar to target data is not sufficient, because our source domain tweets are noisy: tweets with nearly identical content can have contradicting labels. This makes it necessary to check whether tweets contain consistent information about the target data or not before being selected.

3 PROBLEM DEFINITION

Let X be the observable feature space to represent the data in, and Y be the label space: $Y = \{anger, disgust, fear, joy, sadness, surprise\}$. The labeled tweet set (i.e., source domain labeled data) is denoted by $D_l^s = \{(x_i^s, y_i) \in X \times Y \mid y_i$ is the label associated with the instance $x_i^s\}$. Let D_l^t be the target domain (e.g., blogs and fairy tales) labeled data, D_u^t be the target domain unlabeled data, and $D^t = D_l^t \cup D_u^t$ be the overall target domain data. Our objective is: Given a large source domain labeled dataset D_l^s and a target domain labeled dataset D_l^t ($|D_l^s| >> |D_l^t|$), construct a classifier $\hat{c} : X \to Y$ that predicts emotion labels for target domain unlabeled instances.

4 THE PROPOSED APPROACH

We first describe the framework, and then present a scoring function that calculates source instances' informativeness using three factors: *consistency*, *diversity*, and *similarity*. Through informativeness measurement, highly informative source domain tweets will be selected and added to target domain training data so that we can obtain an improved classifier using the enriched training data.

4.1 The Framework

The framework augments target domain labeled data D_l^t with a subset of instances from source domain labeled data D_l^s to improve overall classification accuracy on target domain unlabeled data D_u^t. For this purpose, we first train a classifier using D_l^t and apply it to D_l^s to select a subset of informative instances. If the label of a source domain instance is correctly predicted by the classifier, this instance is regarded as redundant, i.e., the corresponding information is already contained in the target domain instances. If the predicted label is incorrect, then we consider this source domain instance as a candidate for addition, because it may contain information that is lacking in target domain labeled data. A scoring function calculates its informativeness score and decides whether to select it.

The use of informative source instances with D_l^t can yield a new classifier. Ideally, one would expect this new classifier to correctly classify more target domain instances. However, it may misclassify the target domain labeled instances that were correctly classified initially, if a few false informative instances containing inconsistent information were selected. When such misclassification happens, we resort to a "counterbalancing" process to recover. This is achieved by adding these misclassified target domain labeled instances with their correct labels to improve the classification accuracy. In other words, those misclassified target domain labeled instances are given extra weight in training data.

Algorithm 1: The framework

Input: D_l^s, D_l^t, D_u^t, k, δ
Output: Adaptive classifier $\hat{c} : X \to Y$

1 Train an initial classifier c_0 with D_l^t;
2 $T_{correct}^t \leftarrow$ Set of instances from D_l^t that can be correctly classified by c_0;
3 Initialize $T \leftarrow D_l^t$, $T_{info}^s \leftarrow \emptyset$, $T_{wrong}^t \leftarrow \emptyset$, $T^s \leftarrow D_l^s$;
4 **repeat**
5 $\quad T \leftarrow T \cup T_{info}^s \cup T_{wrong}^t$;
6 $\quad$ Train a classifier c with T;
7 $\quad T_{wrong}^s \leftarrow$ Set of instances from T^s that are misclassified by c ;
8 $\quad T_{info}^s \leftarrow$ Top k instances with informativeness score $\phi(\cdot,\cdot)$ greater than δ from T_{wrong}^s;
9 $\quad T^s \leftarrow T^s - T_{info}^s$;
10 $\quad T_{wrong}^t \leftarrow$ Set of instances from $T_{correct}^t$ that are misclassified by c;
11 **until** $\underline{|T_{info}^s| < k}$;
12 **return** $\underline{c}$

Algorithm 1 illustrates the framework. Specifically, the algorithm takes as input D_l^s, D_l^t, D_u^t, a natural number k indicating the number of source instances to be added per iteration, and a real number δ indicating the threshold for selecting source informative instances. The output is an adaptive classifier $\hat{c}$.

We start with training an initial classifier c_0 using D_l^t (line 1). We initialize $T_{correct}^t$ with instances from D_l^t that can be correctly classified by c_0 (line 2). We initialize the overall training data T to D_l^t, newly selected informative source domain instances T_{info}^s to $\emptyset$, counterbalancing target domain instances T_{wrong}^t to $\emptyset$, and source domain candidate instances T^s to D_l^s (line 3).

In every iteration, we first add the newly selected informative instances T_{info}^s and counterbalancing target domain instances T_{wrong}^t into the overall training data T (line 5) that will be used to train a classifier c (line 6). We set T_{wrong}^s to the instances in T^s whose labels are different from those predicted by classifier c (line 7). As discussed earlier, these instances have a potential to augment target domain training data by complementing them with the information that they lack. We then set T_{info}^s to the top k informative instances selected from T_{wrong}^s based on a scoring function that will be explained later (line 8). We remove the newly selected informative source instances T_{info}^s from source domain instances T^s (line 9). If a few false informative instances that contain inconsistent information were selected and added to the training

data, classifier c may misclassify instances in $T^t_{correct}$ that were initially correctly classified by c_0. To counterbalance such an effect, we set T^t_{wrong} to the instances in $T^t_{correct}$ that are misclassified by classifier c (line 10). The instances in T^t_{wrong} will be added to the training data again (i.e., given extra weight) in a new iteration. As we iteratively select informative instances out of T^s, the remaining informative instances in T^s will be reduced. The entire process will terminate when we cannot select sufficient number (k) of instances in an iteration (line 11).

4.2 Selecting Informative Instances

To select informative instances from T^s_{wrong}, we define a source instance's informativeness score as the product of its consistency (λ^c), diversity (λ^d), and similarity (λ^s) factors:

$$\phi(x^s_i, y_i) = \lambda^c(x^s_i, y_i)\ \lambda^d(x^s_i)\ \lambda^s(x^s_i, y_i), \tag{1}$$

so that the instance will achieve a large informativeness score only *when all the three factors are large*. If one factor is small, the informativeness will be penalized after the multiplication. We now show how to calculate each score.

4.2.1 Consistency. Consider the source domain self-labeled tweets are noisy, we want to add a source domain instance that is unambiguously associated with a single label. Moreover, this label should be consistent with the expressed emotion in text. As a counter example, in addition to its single label *fear*, tweet #1 conveys another emotion *joy*. Such instances contain inconsistent information and therefore should not be selected. Specifically, we select instances whose features provide strong support for its label and little support for other emotions, based on D^s_l and D^t_l.

Let $x_a \in X$ be an arbitrary source or target instance, and $y_b \in Y$ be an arbitrary emotion label. We want to construct a **consistency function** $\gamma(x_a, y_b)$ to estimate the confidence of label $y_b \in Y$ being consistent with instance $x_a \in X$, verified using D^s_l and D^t_l. For x_a and all its present features $x_{a,m}$ (i.e., its component words), we define $x_{a,u}$ and $x_{a,v}$ as the strongest supporting features for label y_b according to D^s_l and D^t_l, respectively:

$$x_{a,u} = \arg\max_{x_{a,m}} \{p^s(y_b|x_{a,m})\} \tag{2}$$

$$x_{a,v} = \arg\max_{x_{a,m}} \{p^t(y_b|x_{a,m})\}, \tag{3}$$

where $p^s(y_b|x_{a,m})$ and $p^t(y_b|x_{a,m})$ stand for the conditional probabilities of y_b given $x_{a,m}$ based on D^s_l and D^t_l, respectively. For tweet #1, the strongest supporting features for its label *fear* would be "hope" and "present" as: $p^s(fear|\text{“}hope\text{”}) = 0.509, p^t(fear|\text{“}present\text{”}) = 0.214$.

Similarly, we define $x'_{a,u}$ and $x'_{a,v}$ as the strongest supporting features of x_a for any emotion y'_b other than y_b ($y'_b \in Y \wedge y'_b \neq y_b$), based on D^s_l and D^t_l, respectively:

$$x'_{a,u} = \arg\max_{x_{a,m},\ y'_b} \{p^s(y'_b|x_{a,m})\} \tag{4}$$

$$x'_{a,v} = \arg\max_{x_{a,m},\ y'_b} \{p^t(y'_b|x_{a,m})\}. \tag{5}$$

For tweet #1, the strongest supporting features for any label other than *fear* would be "tmmrw" and "night" as: $p^s(joy|\text{“}tmmrw\text{”}) = 0.563, p^t(joy|\text{“}night\text{”}) = 0.596$. Next, we use the **margin** between the largest conditional probability supporting y_b and that supporting y'_b to define the consistency function as follows:

$$\begin{aligned}\gamma(x_a, y_b) = &\max\left\{p^s(y_b|x_{a,u}), p^t(y_b|x_{a,v})\right\} \\ &- \max\left\{p^s(y'_b|x'_{a,u}), p^t(y'_b|x'_{a,v})\right\},\end{aligned} \tag{6}$$

where a large value indicates that: (1) x_a has a strong supporting feature for its label y_b, i.e., large $p^s(y_b|x_{a,u})$ and/or $p^t(y_b|x_{a,v})$; and (2) according to both D^s_l and D^t_l, the chance that x_a expresses any emotion y'_b other than y_b is small, i.e., both $p^s(y'_b|x'_{a,u})$ and $p^t(y'_b|x'_{a,v})$ are small. The larger the value, the more consistent the label y_b is with the expressed emotion in x_a. A negative value indicates that the label y_b is not the most likely label for x_a. For example, tweet #1 has a negative score: $max(0.509, 0.214) - max(0.563, 0.596) = -0.087$, because besides emotion *fear*, it expresses the emotion *joy* too. Such instances with negative scores are likely to contain inconsistent information and thus are not selected.

Now, we apply the consistency function to measure the **consistency** between source instance x^s_i and its label y_i:

$$\lambda^c(x^s_i, y_i) = \gamma(x^s_i, y_i). \tag{7}$$

4.2.2 Diversity. The measure of diversity emphasizes source domain instances that have distinctive features which are infrequent in the target domain training data. A distinctive feature usually carries effective information to identify the emotion, but if this feature frequently appears in the training data, it may suggest that the target domain already has abundant information about this feature; and therefore adding the instances that contain this feature may not further improve the classifier. Rather than selecting the source instances with the redundant information, it is preferable to select the instances that complement the information that the target domain lacks, i.e., the instances with distinctive features that are less frequent in the training data.

For x^s_i, we apply Equations 2, 3 to find its two supportive features $x^s_{i,u}$ and $x^s_{i,v}$ for its label y_i based on D^s_l and D^t_l, respectively. The most supportive feature is the one with the larger conditional probability out of these two features:

$$x^s_{i,w} = \begin{cases} x^s_{i,u}, & \text{if } p^s(y_i|x_{i,u}) \geq p^t(y_i|x_{i,v}) \\ x^s_{i,v}, & \text{otherwise.} \end{cases} \tag{8}$$

For tweet #1, since $p^s(fear|\text{“}hope\text{”}) \geq p^t(fear|\text{“}present\text{”})$, "hope" is the most supportive feature. If "hope" is infrequent in the training data, we want to promote this tweet to increase the diversity; otherwise, we want to demote this tweet. Let $df(x^s_{i,w})$ be the number of instances that contain feature $x^s_{i,w}$ in the training data T (i.e., the document frequency). We define the diversity of x^s_i using the **exponential decay** of the document frequency of its most supportive feature $x^s_{i,w}$:

$$\lambda^d(x^s_i) = e^{-\theta\ df(x^s_{i,w})}, \tag{9}$$

where θ is a decay constant. The smaller the $df(x^s_{i,w})$, the larger the diversity with a max value of 1. In the extreme case of $df(x^s_{i,w}) = 0$, this feature is source domain-specific and does not occur in the target domain at all (e.g., slangs), and we instead use the next most supportive feature that is present in the target domain.

4.2.3 Similarity. Prior studies [10, 19] have shown that the adaptability of machine translation models can be improved by selecting source domain sentences that are similar to target domain sentences, because these sentences can better match the test data in the target domain. In this paper, besides the content similarity, we also need to examine the label similarity. Otherwise, we may select source instances (tweets) with nearly identical content but labeled with different emotion hashtags by the authors. Both tweet #2 and #3 in Table 1 describe similar scenarios involving a phone's running out of battery, but they are labeled with *sadness* and *anger*, respectively. Moreover, we emphasize the unlabeled instances that classifier c is uncertain about, and select source instances $(x_i^s, y_i) \in T^s$ that are similar to them. When c is uncertain about $x_j^t \in D_u^t$, it suggests that the target domain is lacking the corresponding information to make a confident prediction.

Specifically, to encourage the selection of source instances that share high content and label similarities with target domain unlabeled instances that classifier c is uncertain about, we define the similarity factor of x_i^s as:

$$\lambda^s(x_i^s, y_i) = \max_{x_j^t \in D_u^t} \{\pi^c(x_i^s, x_j^t)\ \pi^l(x_j^t, y_i)\ \pi^u(x_j^t)\}, \quad (10)$$

where $\pi^c(x_i^s, x_j^t)$ denotes the content similarity between x_i^s and x_j^t, $\pi^l(x_j^t, y_i)$ indicates how likely x_j^t and x_i^s share the same label y_i, and $\pi^u(x_j^t)$ represents the uncertainty of classifier c regarding x_j^t. To quantify the **content similarity** between x_i^s and x_j^t, we apply cosine similarity to their weight vectors $\overrightarrow{V^s}(x_i^s)$ and $\overrightarrow{V^t}(x_j^t)$:

$$\pi^c(x_i^s, x_j^t) = \frac{\overrightarrow{V^s}(x_i^s) \cdot \overrightarrow{V^t}(x_j^t)}{|\overrightarrow{V^s}(x_i^s)||\overrightarrow{V^t}(x_j^t)|}. \quad (11)$$

The purpose of weight vector representation is to boost the weights of important words, so that x_i^s and x_j^t are similar to each other only when they share important words. For $(x_i^s, y_i) \in T^s$, we want to assign larger weights to words that are strong indicators of its label y_i. For its m-th present feature $x_{i,m}^s$, we apply conditional probability of y_i, given this feature based on D_l^s, as its weight:

$$weight_{i,m}^s = p^s(y_i | x_{i,m}^s). \quad (12)$$

For a target domain unlabeled instance $x_j^t \in D_u^t$, we cannot apply the above equation to calculate the conditional probability of the label given a feature because its label is unknown. Thus, we use a TF-IDF weighting scheme to assign weights instead. Since we are conducting sentence-level emotion identification and most features usually occur once in a sentence, we skip the TF and apply only the prob IDF from SMART notation [20]. Specifically, the weight of the n-th present feature $x_{j,n}^t$ of the instance x_j^t is:

$$weight_{j,n}^t = \max\left\{0, \log_{10} \frac{N - df(x_{j,n}^t)}{df(x_{j,n}^t)}\right\}, \quad (13)$$

where $N = |T|$ and $df(x_{j,n}^t)$ is the number of instances that contain feature $x_{j,n}^t$ in training data T.

Besides the content similarity, we also need to consider the **label similarity**. Otherwise, we may add instance x_i^s that is similar to x_j^t in content but has a contradicting label. Since the label of x_j^t is yet to be predicted, we cannot directly compare the labels of x_j^t and x_i^s. Instead, we estimate how likely is x_j^t to share the same emotion label y_i of x_i^s. For x_j^t, we apply the consistency function (Equation 6) to measure the confidence that x_j^t has the same label y_i as x_i^s:

$$\pi^l(x_j^t, y_i) = \gamma(x_j^t, y_i). \quad (14)$$

The larger the value, the more likely that x_j^t and x_i^s share the same label y_i. When the value is negative, it is likely that the label of x_j^t is different from that of x_i^s.

Let y_j^* be the most likely label predicted by c for x_j^t. We define the **uncertainty** of classifier c regarding x_j^t as:

$$\pi^u(x_j^t) = 1 - p(y_j^* | x_j^t; c). \quad (15)$$

In summary, the informativeness scoring function achieves a large value when all the following three conditions are satisfied for a source instance: (1) its label is consistent with its content, (2) it contains a distinctive feature that is infrequent in target training data, and (3) it is similar to a target domain unlabeled instance whose label cannot be predicted by the classifier c with confidence.

Table 2: Dataset statistics

	Source Domain	*Target Domains*			
Name	Twit	Blog	Diary	Exp	Fairy
Instance #	100,000	1,290	507	384	1,722

5 EXPERIMENTS

For the source domain data, we used the emotion hashtags in [35] as filtering keywords, and collected 100K emotion tweets as the source data **Twit**. We used four sentence-level multi-class emotion datasets as target domain data: **Blog** [2], **Diary** [24], **Exp** [23] (sentences describing personal experiences), and **Fairy** [1]. These datasets have different emotion classes, which brings extra complexity to the experiments. For example, Diary has the emotions *interest* and *shame*, but Blog does not. To concentrate on the adaptation problem, we focus on emotion classes which are common to every dataset: *anger*, *disgust*, *fear*, *joy*, *sadness*, and *surprise*. The sizes of all the datasets are shown in Table 2. We observe that these datasets are relatively small compared with Twit (100K) – Diary and Exp contain about 500 sentences. We believe the proposed algorithm that exploits Twitter data could be helpful for emotion identification in these cases.

We performed the same data preprocessing on all the datasets. Specifically, we lower-cased all the words; replaced letters/punctuation marks that are repeated with the same two letters/punctuation marks (e.g., "cooool" → "cool", "!!!!!" → "!!"); and normalized some frequently used informal expressions (e.g., "ll" → "will"). For Twit, we replaced user mentions (e.g., "@Justin") with "@user" to anonymize users; and stripped hash symbols ("#today" → "today").

We used a logistic regression classifier in LIBLINEAR [11] for classification because: (1) it is very fast, and (2) it natively supports probability output that is used to calculate uncertainty in Equation

15. We experimented with different feature representations: unigrams, bigrams, unigrams and bigrams, and found that the unigram representation achieves the best performance for all the baseline approaches. So we report results using unigrams. We performed frequency-based feature selection: the unigrams appearing in at least five different tweets in Twit or at least two different sentences in other datasets were selected as features. For each dataset, we applied five-fold cross validation where four folds were used as target domain labeled data and the remaining fold was used as test data. We repeated this five times and the average of micro-averaged F_1 scores in five folds was used for performance measurement.

We use **CDS** to abbreviate the proposed method . To evaluate the contribution of each factor, we use **C**, **D**, and **S** to abbreviate three variants of CDS by using only **C**onsistency, only **D**iversity and only **S**imilarity factors, respectively. We set exponential decay constant $\theta = 0.05$. We set number of selected informative instances per iteration $k = 0.05\ |D_l^t|$. That is 5% of the labeled instances in target domains. We empirically set informativeness threshold $\delta = 0.0005$ as its default value. We will study the effect of changing k and δ later. We used add-0.5 smoothing [20] to estimate the conditional probabilities of a label given a feature: e.g., $p^s(y_i|x^s_{i,m})$ and $p^t(y_i|x^s_{i,m})$.

5.1 Baseline Approaches

Most of existing studies on domain adaptation focus on binary sentiment classification and they are not applicable for our multi-class emotion classification problem [5, 14, 26]. Therefore, we compare CDS against the following five approaches that support multi-class classification-based domain adaptation instead. Need to mention that prior studies [4, 9] find most of the following baselines surprisingly difficult to beat.

Source Only (SO): Without any adaptation, we directly apply the classifier trained on source Twit to target datasets.

Target Only (TO): Since the target domain training data is more representative of target domains than Twit is, we train classifiers using only the target domain training data.

Feature Augmentation (FA): The idea is to "augment the feature space of both the source and target data and use the result as input to a standard learning algorithm" [9]. After feature augmentation, the new feature space contains three sub-spaces: source domain specific feature space, target domain specific feature space and source-target-domain overlapping feature space. In the process of training on the combination of the source and target domain training data, the classifier can select and apply distinctive features from the augmented feature space.

Feature Injection (FI): The idea is to first train a source classifier using only the source data. Then, this classifier is applied to both the labeled and unlabeled data in the target domain, and its probability outputs (i.e., the probabilities of x_j expressing different emotions) will be injected as additional features. A target classifier will be trained using the target data after feature injection [9].

Balance Weight (BW): Given that labeled instances in the target domain are more representative of the target domain than the source instances, the idea is to assign larger weights for the target instances so that the weighted sum of target instances equals that of source instances [16]. The weight of every instance in D_l^t is set to $\frac{|D_l^s|}{|D_l^t|}$, and then a classifier is trained on $D_l^s \cup D_l^t$.

Table 3: Results for all approaches on four target datasets. For each row, the best approach is in bold, the second best is underlined, and the third best is under-waved.

	Micro-averaged F_1								
Datasets	SO	TO	FI	FA	BW	C	D	S	CDS
Blog	0.5054	0.6488	0.6930	0.6969	0.6922	0.6984	0.6868	0.6915	**0.7008**
Diary	0.4870	0.4910	0.5423	0.5383	0.5621	0.5816	0.5246	0.5955	**0.6092**
Exp	0.5261	0.5053	0.5729	0.5834	0.6379	0.6379	0.5598	0.6691	**0.6899**
Fairy	0.4210	0.6574	0.6684	0.6754	0.6702	0.6707	0.6527	0.6701	**0.6812**
Average	0.4849	0.5756	0.6191	0.6235	0.6404	0.6472	0.6060	0.6566	**0.6703**

5.2 Evaluations on Domain Adaptation

Table 3 presents the experimental results in micro-averaged F_1 metric obtained by all approaches on four datasets. We observe that: (1) in descending order of the averages of their micro-averaged F_1 across all datasets, these approaches rank as follows: CDS (0.6703), S (0.6566), C (0.6472), BW (0.6404), FA (0.6235), FI (0.6191), D (0.6060), TO (0.5756), SO (0.4849); (2) CDS outperforms all the baseline approaches on every dataset; however, the difference between CDS and BW on F_1 metric is not statistically significant (with p-values in parenthesis): Blog(0.204), Diary(0.151), Exp(0.092), Fairy(0.164); part of the reason could be the high variance caused by the relatively small number of target domain instances in experiments. (3) Among the component factors, Similarity (0.6566) performs the best, followed by Consistency (0.6472) and Diversity (0.6060). (4) [9] finds that FA performs worse when the source and target domains are very similar (i.e., SO performs similar to or better than TO); In contrast, if the source and target domains are different (i.e., SO performs worse than TO), FA tends to outperform other approaches. This is corroborated in our experiment: FA performs the best among all the baselines on Blog and Fairy, where as SO performs worse than TO. Lastly, (5) BW outperforms other baselines on Diary and Exp, where the performance of SO is similar to or better than that of TO. This seems to suggest that BW complements FA on datasets where the source and target domains are very similar.

The key difference between CDS and the baseline is that CDS doesn't use all the tweets from the self-labeled noisy tweets. Instead, it selects the most informative ones that have more potential to boost emotion classification in target domains. The results that CDS consistently outperforms baselines suggest that the combination of consistency, diversity and similarity is effective at selecting the informative tweets for cross-domain emotion identification.

5.2.1 Influence of the Parameters: k, δ. We vary, k, the number of selected informative instances per iteration, from $0.05\ |D_l^t|$ to $0.5\ |D_l^t|$ (i.e., 5% to 50%), to show how it impacts results in Figure 1. The general trend is that micro-averaged F_1 slowly decreases as we select more instances per iteration across all the datasets, because target labeled data gets diluted faster with source data. The best result in micro-averaged F_1 is achieved when the proportion is 0.05.

The informativeness score (Equation 1) can be negative under two conditions: either consistency $\lambda^c(x_i^s, y_i) < 0$ or label similarity

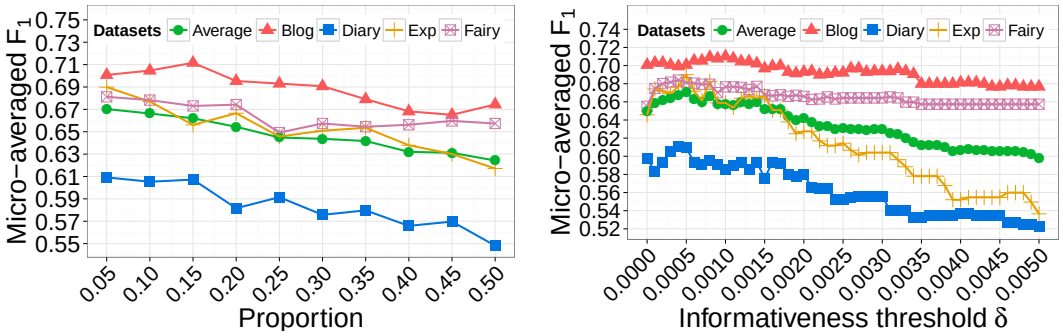
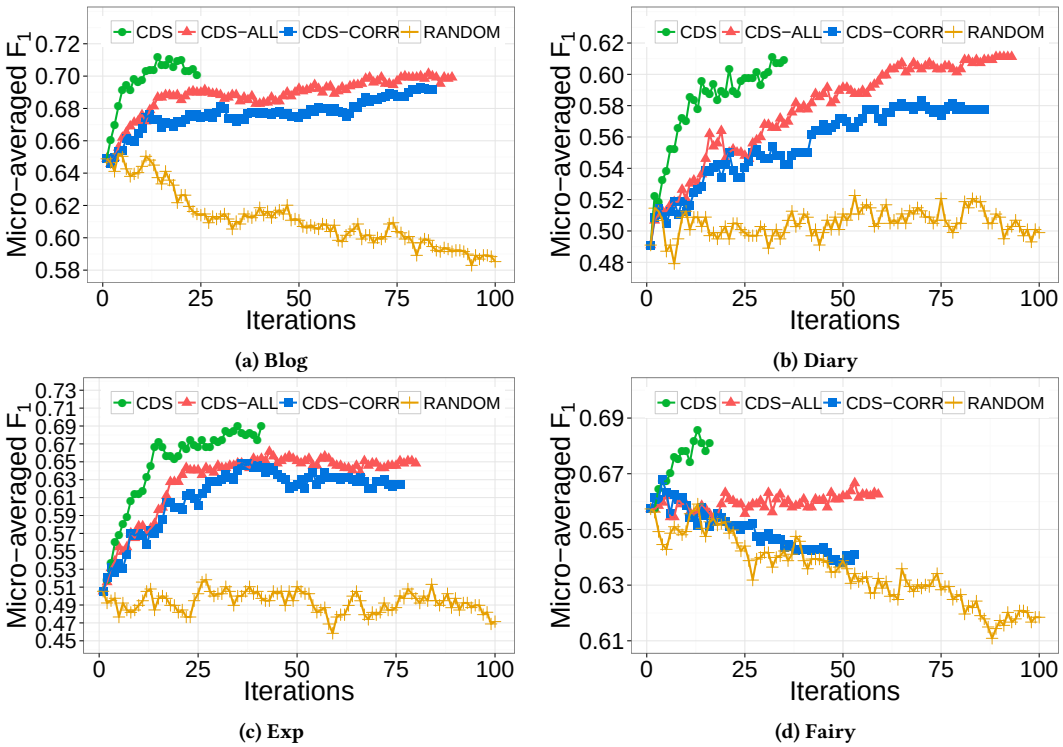


Figure 1: Influence of the Parameters. Left: varying the number of selected informative instances (k). Right: varying the informativeness threshold (δ)



(a) Blog **(b) Diary**

(c) Exp **(d) Fairy**

Figure 2: Results of applying different strategies to select informative instances on four datasets.

$\pi^l(x_j^t, y_i) < 0$. In practice, we skipped the instances that satisfy either condition because such instances are likely to contain inconsistent information. We increase the informativeness threshold δ from 0 and show how it influences results in Figure 1. When we increase δ from 0 to 0.0002, the average of micro-averaged F_1 increases from 0.6498 to 0.6621, because we are selecting better tweets of larger informativeness. When δ is between 0.0002 and 0.0008, the average of micro-averaged F_1s on all datasets is ≥ 0.66. When we increase δ beyond 0.0008, the general trend is that F_1 starts decreasing on almost all the datasets, while decreasing faster on Diary and Exp. By further increasing δ, we make the bar for selecting informative tweets so high that we cannot obtain enough informative tweets. It is important to mention that the reason δ is very small is that δ is the multiplication of several small factors (refer to Equations 1, 10).

5.2.2 Evaluations of Instance Selection Strategies. We evaluate the strategies for the selection of informative instances to show the effectiveness of selecting instances out of T^s_{wrong} (instead of T^s). We define several variants of CDS with the following changes: **CDS-ALL** selects instances from T^s; **CDS-CORR** selects instances from T^s that are *correctly* classified by c. **RANDOM** is a baseline approach that randomly selects instances from T^s during each

iteration. We let each approach run up to 100 iterations and the result remains at the value of the last iteration unless one approach meets the stopping condition early.

We show the results of applying these four selection strategies in Figure 2. In descending order of the micro-averaged F_1, the strategies rank as follows: CDS, CDS-ALL, CDS-CORR, Random, which is consistent across all datasets, with the exception of CDS-ALL (0.6112) performing marginally better than CDS (0.6092) on Diary. Among all the strategies, CDS improves F_1 with the least number of iterations. The reason why CDS improves F_1 faster than CDS-ALL and CDS-CORR is that we feed CDS with instances from T^s_{wrong} which are incorrectly classified by classifier c. Some of these instances contain information that is lacking in the target domain. Since the input of CDS-ALL is a super set of CDS, it usually achieves similar results in the end, but it takes far more iterations for CDS-ALL to terminate. The RANDOM strategy is not a good choice because it results in performance declines on Blog, Exp, and Fairy, and barely improves the performance on Diary.

6 CONCLUSIONS

We studied the problem of leveraging self-labeled noisy Twitter data to improve emotion identification across different domains via adaptive instance selection. We proposed a framework that iteratively selects tweets that are informative about target domains using criteria based on three carefully designed measures: consistency, diversity, and similarity. This approach has the following advantages: (1) Unlike most of the prior work that support only binary cross-domain sentiment classification, it supports multi-class fine-grained cross-domain emotion identification. (2) It can consume self-labeled noisy tweets to select the most informative ones to improve target domain emotion identification in an adaptive and progressive way. (3) Extensive experiments on four target domains show that our approach is effective for cross-domain emotion identification and consistently outperforms baseline approaches.

ACKNOWLEDGMENTS

We acknowledge partial support from the National Science Foundation (NSF) award: CNS-1513721: "Context-Aware Harassment Detection on Social Media" and the National Institutes of Mental Health (NIH) award: 1R01MH105384-01A1: "Modeling Social Behavior for Healthcare Utilization in Depression." Points of view or opinions in this document are those of the authors and do not necessarily represent the official position or policies of the NSF or NIH. Many thanks to Dr. Ramakanth Kavuluru for several suggestions improving the presentation of technical details of this paper.

REFERENCES

[1] Cecilia Ovesdotter Alm, Dan Roth, and Richard Sproat. 2005. Emotions from text: machine learning for text-based emotion prediction. In HLT and EMNLP. ACL, 579–586.
[2] Saima Aman and Stan Szpakowicz. 2008. Using Roget's Thesaurus for Fine-grained Emotion Recognition.. In IJCNLP. 312–318.
[3] Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In EMNLP. 355–362.
[4] Plank Barbara. 2011. Domain adaptation for parsing. Ph.D. Dissertation. University of Groningen. Advisor(s) Noord, Gertjan van.
[5] John Blitzer, Mark Dredze, Fernando Pereira, and others. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In ACL, Vol. 7. 440–447.
[6] Felipe Bravo-Marquez, Eibe Frank, Saif M Mohammad, and Bernhard Pfahringer. 2016. Determining word–emotion associations from tweets by multi-label classification. In WI'16. IEEE Computer Society, 536–539.
[7] Keke Chen, Rongqing Lu, CK Wong, Gordon Sun, Larry Heck, and Belle Tseng. 2008. Trada: tree based ranking function adaptation. In CIKM. ACM, 1143–1152.
[8] W. Dai, Q. Yang, G.R. Xue, and Y. Yu. 2007. Boosting for transfer learning. In ICML. ACM, 193–200.
[9] H. Daumé. 2007. Frustratingly easy domain adaptation. In ACL, Vol. 45. 256.
[10] Matthias Eck, Stephan Vogel, and Alex Waibel. 2004. Language Model Adaptation for Statistical Machine Translation Based on Information Retrieval.. In LREC.
[11] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. JMLR 9 (2008), 1871–1874.
[12] George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In EMNLP. ACL, 451–459.
[13] Diman Ghazi, Diana Inkpen, and Stan Szpakowicz. 2010. Hierarchical versus flat classification of emotions in text. In NAACL HLT workshop on computational approaches to analysis and generation of emotion in text. ACL, 140–146.
[14] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In ICML. 513–520.
[15] Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In EAMT, Vol. 2005. 133–142.
[16] J. Jiang and C.X. Zhai. 2007. Instance weighting for domain adaptation in NLP. In ACL, Vol. 45. 264.
[17] Michel Klein, Adnan Manzoor, Julia Mollee, and Jan Treur. 2014. Effect of changes in the structure of a social network on emotion contagion. In WI and IAT. IEEE Computer Society, 270–277.
[18] Gilly Leshed and Joseph'Jofish' Kaye. 2006. Understanding how bloggers feel: recognizing affect in blog posts. In CHI. 1019–1024.
[19] Yajuan Lü, Jin Huang, and Qun Liu. 2007. Improving Statistical Machine Translation Performance by Training Data Selection and Optimization.. In EMNLP-CoNLL. 343–350.
[20] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. Introduction to information retrieval. Vol. 1. Cambridge university press.
[21] Lionel Martin and Pearl Pu. 2014. Prediction of helpful reviews using emotions extraction. In AAAI.
[22] Saif M Mohammad. 2012. # Emotional tweets. In *SEM. ACL, 246–255.
[23] Alena Neviarouskaya, Helmut Prendinger, and Mitsuru Ishizuka. 2010. Recognition of affect, judgment, and appreciation in text. In COLING. ACL, 806–814.
[24] Alena Neviarouskaya, Helmut Prendinger, and Mitsuru Ishizuka. 2011. Affect analysis model: novel rule-based approach to affect sensing from text. NLE 17, 1 (2011), 95–135.
[25] Pedro A Nogueira, Rúben Aguiar, Rui Rodrigues, and Eugénio Oliveira. 2014. Computational Models of Players' Physiological-Based Emotional Reactions: A Digital Games Case Study. In WI and IAT, Vol. 3. IEEE, 278–285.
[26] S.J. Pan, X. Ni, J.T. Sun, Q. Yang, and Z. Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In WWW. ACM, 751–760.
[27] S.J. Pan and Q. Yang. 2010. A survey on transfer learning. TKDE 22, 10 (2010), 1345–1359.
[28] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, Qiang Yang, and others. 2011. Domain adaptation via transfer component analysis. IEEE Transactions on Neural Networks 22, 2 (2011), 199–210.
[29] Matthew Purver and Stuart Battersby. 2012. Experimenting with distant supervision for emotion classification. In EACL. ACL, 482–491.
[30] Kirk Roberts, Michael A Roach, Joseph Johnson, Josh Guthrie, and Sanda M Harabagiu. 2012. EmpaTweet: Annotating and Detecting Emotions on Twitter.. In LREC. 3806–3813.
[31] Carlo Strapparava and Rada Mihalcea. 2008. Learning to identify emotions in text. In SAC. ACM, 1556–1560.
[32] Jared Suttles and Nancy Ide. 2013. Distant supervision for emotion classification with discrete binary values. In CICLing. Springer, 121–136.
[33] Ryoko Tokuhisa, Kentaro Inui, and Yuji Matsumoto. 2008. Emotion classification using massive examples extracted from the web. In COLING. ACL, 881–888.
[34] Quang Hong Vuong and Atsuhiro Takasu. 2014. Transfer Learning for Emotional Polarity Classification. In WI and IAT. IEEE Computer Society, 94–101.
[35] Wenbo Wang, Lu Chen, Krishnaprasad Thirunarayan, and Amit P Sheth. 2012. Harnessing twitter "big data" for automatic emotion identification. In PASSAT and SocialCom. IEEE, 587–592.
[36] Chayanin Wong and In-Young Ko. 2016. Predictive Power of Public Emotions as Extracted from Daily News Articles on the Movements of Stock Market Indices. In WI. IEEE, 705–708.
[37] R. Xu, J. Xu, and X. Wang. 2011. Instance level transfer learning for cross lingual opinion analysis. In the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis. ACL, 182–188.