Discovering and Controlling for Latent Confounds in Text Classification Using Adversarial Domain Adaptation

Virgile Landeiro[*] Tuan Tran[*] Aron Culotta[*]

Abstract

In text classification, the testing data often systematically differ from the training data, a problem called *dataset shift*. In this paper, we investigate a type of dataset shift we call *confounding shift*. Such a setting exists when two conditions are met: (a) there is a confound variable Z that influences both text features X and class label Y; (b) the relationship between Z and Y changes from training to testing. While recent work in this area has required confounds to be known ahead of time, this is unrealistic for many settings. To address this shortcoming, we propose a method both to discover and to control for potential confounds. The approach first uses neural network-based topic modeling to discover potential confounds that differ between training and testing data, then uses adversarial training to fit a classification model that is invariant to these discovered confounds. We find the resulting method to improve over state-of-the-art domain adaptation method, while also producing results that are competitive with those obtained when confounds are known ahead of time.

1 Introduction

In text classification, the testing data is often drawn from a different distribution than the training data. This *dataset shift* may happen because of a change in domain (e.g., from movie reviews to product reviews) or because of data drift over time (e.g., words that predict conservative political views in 2014 may predict liberal political views in 2018). *Domain adaptation* (DA) algorithms learn classifiers in the presence of such dataset shift. Most DA approaches analyze the differences between the training and testing data in order to learn domain-invariant feature representations [2, 6, 3].

In this paper, we investigate a specific type of dataset shift we call *confounding shift*. A confounding variable, or confound, is a latent variable Z that influences both the text features X and the class label Y. Confounds in text classification have received attention recently due to their prevalence in computational social science as well as their importance in algorithmic fairness in machine learning [9, 12, 14, 13]. Confounding shift occurs when the relationship between the confound Z and class label Y changes from training to testing. For example, consider a training set for movie review sentiment classification in which horror movies are overall more likely to have negative reviews than other genres. Here, movie genre confounds sentiment classification. However, if the testing set contains many critically acclaimed horror films, then the classifier may have a high false negative rate for them due to the presence of horror-related terms.

Existing approaches to controlling for confounds in text classification assume the set of confounds are known *a priori* [9, 14, 13]. However, in real-world settings with streaming data, we will rarely know the confounds in advance, limiting the applicability of these approaches.

In this paper, we propose a method to both discover and control for confounds in text classification, thereby improving robustness to confounding shift. The approach first builds on recent work in adversarial domain adaptation [6] to fit a neural network-based topic model that discovers latent variables correlated with the input X but not with the target variable y or with the source of the data s (train or test). This allows us to find latent variables that change from training to testing time. Our assumption is that these words are correlated with the latent confounding variable. Next, to control for these variables, we fit a classifier with an adversarial training objective that learns representations that do a poor job of predicting the discovered latent variables. The resulting approach can be understood as a type of unsupervised domain adaptation tailored to the task of confounding shift.

We evaluate the approach on three text classification tasks where there is high confounding shift between training and testing due to one latent confounding variable. We find that 1) state of the art domain adaptation recovers well from high confounding shift and 2) when the latent confounding variable is well captured by a topic model, the approach we propose is able to outperform domain adaptation; otherwise it is on par with domain adaptation. In the remainder of the paper, we summarize related work in Section 2, formalize our problem setting in Section 3, and describe our proposed approach in Section 4. Then, we describe the data and experimental results in Sections 5 and 6, respectively. Finally, we discuss the limitations of the approach in Section 7 before giving concluding remarks in Section 8.

2 Related Work

Goodfellow et al. [7] popularized adversarial training for deep generative modeling; Ganin et al. [6] subsequently

[*] Department of Computer Science, Illinois Institute of Technology, Chicago, IL primary contact: vlandeir@hawk.iit.edu

extended this approach to unsupervised domain adaptation. This method fits a neural network using two error signals: (1) the traditional classification error; and (2) the error of predicting whether an example is drawn from the training or testing set. By learning a representation that has low error for (1) and high error for (2), we can hope to learn a representation that is shared by both the training and testing data.

A number of recent approaches applied adversarial training to control for sensitive variables in classification [4, 16, 11, 17]. Other approaches use simpler statistical adjustment methods to remove confounds from text classification algorithms [9]. However, these methods all assume the confounding variables are known in advance.

Our work is inspired in part by Pryzant et al. [13], who learn a lexicon while controlling for given confounds. One of their methods uses adversarial learning to train a classifier that does poorly at predicting a pre-specified confound. In our work, we also use adversarial learning to control for confounds; however, we do not assume we know confounds in advance. Instead, we use latent variable discovery to first identify candidate confounds, then control for them with adversarial learning.

Another more distantly related line of work is on concept drift adaptation, which considers how to train classifiers in streaming settings in order to be robust to dataset shifts [10, 5]. While similar in motivation, this prior work typically assumes an oracle is available to annotate new documents for training, which is not the case in our setting.

In summary, our approach combines two lines of research: domain adaptation and controlling for confounds in text classification. Whereas prior work in controlling for confounds assumes the variables are known in advance, here we discover them. Additionally, while prior work in domain adaptation is designed to learn generic, domain-invariant representations, here we instead tailor domain adaptation methods to the problem of confounding shift.

3 Problem Definition

In common text classification tasks, a training dataset of N instances $D_t = \{(\vec{x}_i, y_i)\}_{i=1}^N$ is available at time t through data collection and annotation. For a vocabulary of size d and a target accepting K different values, each data point is represented as a tuple of the encoded text input $\vec{x}_i \in \mathbb{R}^d$ and the associated target variable $y_i \in \{1, \ldots, K\}$. A predictive model is then fit on the N training instances, resulting in a function $f : \mathbb{R}^d \to \{1, \ldots, K\}$ that is able to assign a label to a new instance. At a later time t^+, new data is collected and labeled using f such that $D_{t^+} = \{(\vec{x}_j, f(\vec{x}_j))\}_{j=1}^M$. In theory, if the data source is the same at times t and t^+, then the distributions $p_t(\vec{x}, y)$ and $p_{t^+}(\vec{x}, y)$ should be the same. However, in text classification, this is often not the case, as languages evolve and some latent factors can impact the joint distributions over words and labels.

Notation	Description
X	Input variables (text features).
Y	Output variable (class label).
Z	Confounding variable.
$\rho(A, B)$	Pearson correlation between two variables A and B.
$P_t(e)$	Probability of event e at time t.
HD(p, q)	Hellinger distance between two probability distributions p and q.

Table 1: Table of notations used across this paper.

In this paper, we focus on the effect of variables that are correlated with both the input $\vec{x}$ and the output y of a text classifier f: confounding variables. In particular, we study the problem of confounding shift that appears when the influence of such a confounding variable z is changing between time t and time t^+. Indeed, if the influence of z does not change between fitting time and prediction time as assumed in classical text classification, then there should be no impact on the performance of f. However, if for instance the influence of z on y increases between t and t^+, then z may introduce spurious correlation, leading to a significantly degraded performance of classifier f on new data. In prior work [9, 13], it is assumed that the confounding variable is known at training time. Here, we propose a two step approach aiming to control for confounding shift caused by one confounding variable:

1. CONFOUNDDISCOVERY tackles the problem of detecting which variables are prone to be a source of confounding shift. The inputs of this subproblem are the dataset at fitting time D_t and the dataset at prediction time D_{t^+}. The output generated by this task is a pair $(\hat{z}_i, \sigma_i)$ for every document in the training dataset, where $\hat{z}_i$ is the estimated value of the confounding variable and σ_i is the confidence of our model that $\hat{z}_i$ is the correct value.

2. CONFOUNDCONTROL uses the output of the previous subproblem to build a classifier g that is robust to confounding shift. Multiple approaches can be applied for this task. In the following section, we explain the adversarial method borrowed from the domain adaptation field we used to solve this problem.

4 Methods

In this section, we first present our method to solve the CONFOUNDDISCOVERY task when an unknown confounding variable z generates confounding shift between times t and t^+. Then, we present an adversarial approach to solve the ConfoundControl problem. Finally, we explain our proposed method as well as multiple baselines.

4.1 Confound Discovery.

First, we fit a topic model on both D_t and D_{t^+} that does not discriminate on the data source s (D_t vs D_{t^+}) nor on the target variable y. We then look for terms that display a change in their distribution over topics between times t and t^+. Since we constrain our topic model to produce topics that are not discriminative of y or s, we hypothesize that the changes in distributions are due to the latent confounding variable as the effect of the other covariates has been dampened by the fact that the topics created are not discriminative. Finally, we use these discovered words to infer a confounding variable $\hat{z}_i$ and a confidence indicator σ_i for each instance in the training set D_t.

Topic modeling. Our topic model is based on the ProdLDA model introduced by Srivastava and Sutton [15]. ProdLDA and LDA [1] are alike in that they are topic models and return similar topic distributions but the former has two main advantages: 1) it is based on neural networks and can therefore be more easily adapted to new problem settings and 2) it can make use of GPUs to speed up computation. The standard architecture of a ProdLDA model is shown in black in Figure 1; it consists of an encoder network (from X to T) followed by a decoder network (from T to $\hat{X}$). It is also very similar to the variational autoencoder architecture [8]; the main difference is that the encoder loss of ProdLDA constrains the latent units to be distributions over documents.

In order to build a topic model that is not discriminative with respect to y and s, we extend the ProdLDA model with the gradient reversal layers proposed by Ganin et al. [6] and indicated by the dashed arrows labeled GR in Figure 1. To do so, we first add two output nodes and their associated discriminative loss to the original ProdLDA model, one for the data source $s \in \{t, t^+\}$ and one for the target variable y. Then, we implement the gradient reversal step such that the partial gradients computed at nodes s and y are multiplied by -1 before they are back propagated to T. This causes the latent units in T to contain representations that are not predictive of s or y.

Note that we do not have label information for data in D_{t^+} so we ignore the y-loss when $s = t^+$, allowing us to train the model in one pass:

$$\begin{aligned} h_1 &= softplus(W_0 X) \\ h_2 &= softplus(W_1 h_1) \\ \mu_T &= BatchNorm(W_2^0 h_2) \\ \sigma_T &= exp(BatchNorm(W_2^1 h_2)) \end{aligned}$$

Using the reparameterization trick similar to that in a Variational Autoencoder [8], we first sample ϵ such that $\epsilon \sim \mathcal{N}(0, I)$. We can then sample T from $\mathcal{N}(\mu_T, \sigma_T)$ and finish the pass through the decoder:

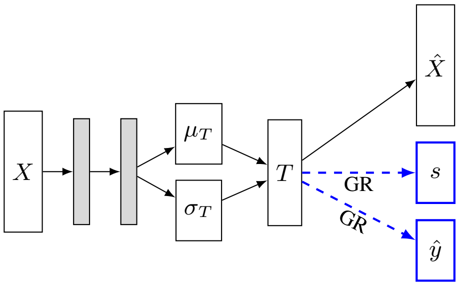

Figure 1: Adversarial and supervised ProdLDA. GR indicates a gradient reversal layer.

$$\begin{aligned} T_{sampled} &= \mu_T + \sigma_T^{1/2} \times \epsilon \\ T &= softmax(T_{sampled}) \\ \hat{X} &= softmax(BatchNorm(W_3^0 T)) \\ s &= softmax(BatchNorm(GR(W_3^1 T))) \\ y &= softmax(BatchNorm(GR(W_3^2 T))) \end{aligned}$$

As discussed by Srivastava and Sutton [15], the objective $L_{\hat{X},T}$ to minimize of the original ProdLDA architecture is the sum of the latent loss (encoder) and the reconstruction loss (decoder). Since we extend the original ProdLDA with two more output nodes, we introduce to the model two additional categorical cross-entropy losses, resulting in the total loss as follows:

$$\begin{aligned} L_{\hat{y}} &= \sum_{i=1}^{N} y_i \log \hat{y}_i \\ L_s &= \sum_{i=1}^{N+M} s_i \log \hat{s}_i \\ L_{total} &= L_{\hat{X},T} + L_{\hat{y}} + L_s \end{aligned}$$

Building a confounding indicator. Once our topic model has been trained, we can obtain the topic distribution $\vec{\tau}_i = \{\tau_{i,1}, \ldots, \tau_{i,l}\}$ for a document $\vec{x}_i$, where l is the number of topics. We combine this information with the encoding of each document to create a topic distribution at the word level. Specifically, for word w_j, we compute $\vec{\omega}_j(D) = \{\omega_{j,1}(D), \ldots, \omega_{j,l}(D)\}$ where

$$\omega_{j,k}(D) = \sum_{\vec{x}_i \in D} x_{i,j} \times \tau_{i,k}$$

where $x_{i,j}$ is the encoding of word w_j in document $\vec{x}_i$ and $\tau_{i,k}$ is the probability of document x_i to be in topic k. To find words that change at the topic level between training and testing, we compute the Hellinger distance $h(w_j)$ for every term such that:

$$h(w_j) = \text{HD}(\omega_j(D_t), \omega_j(D_{t+}))$$

Our hypothesis is that the higher $h(w_j)$ is, the more likely term w_j is to be correlated with the latent confounding variable. Unfortunately, it is not enough to know that a variable is correlated with the confounding variable z, we also need to know if it is positively or negatively correlated with z. Although we do not have access to z, we know that $\rho(y, z)$, the Pearson correlation between y and z, is large because we are in a situation of high confounding shift, therefore we can use $\rho(w_j, y)$ as a noisy proxy for $\rho(w_j, z)$. Using this information, we obtain a score for every document $x_i \in D_t$ indicating how correlated to z a document is and in which direction:

$$h(\vec{x}_i) = \sum_{j \,:\, x_{i,j} \neq 0} \text{sign}(\rho(w_j, y)) \times h(w_j)$$

That is, we sum over each word present in $\vec{x}_i$, multiplying its sign by its Hellinger distance.

Finally, we break down this score in two parts and we let $\hat{z}_i = \text{sign}(h(\vec{x}_i))$ be our estimation of z_i and $\sigma_i = |h(\vec{x}_i)|$ the confidence we have in our estimation.

4.2 Confound Control. We now have a method to detect words that are correlated with the latent confounding variable. We can assign an estimation $\hat{z}$ of this confounding variable to each document in the training set along with σ, a measure of how confident our method is in the estimation. In this section, we present the confounding control methods that we use for our experiments. This approach is inspired by the Domain-Adversarial Neural Network introduced in by Ganin et al. [6] and used with confounding variables by Pryzant et al. [13]. The original model is used for domain adaptation purposes where it uses an adversarial approach to learn representations that are predictive of the class label y while at the same time performing poorly to distinguish between the different domains s. To do so, the authors implemented the neural network displayed in black in Figure 2 where X is the text input, y is the class label, s is the domain and e is a shared representation. The trunk (from X to e) and the first branch (from e to y) creates a standard feed forward network. However, the second branch (from e to s) is not standard as it implements the adversarial part of this network. A gradient reversal layer is added: it is idle during the feed-forward phase but reverses the gradient right before the shared representation e during the back-propagation phase. Doing so will create a shared representation e predictive of y but unable to discriminate amongst domains s, hence leading to a classifier that generalizes to multiple domains. Here, we propose to extend this model by adding a third branch (in blue in Figure 2) that will encourage the shared layer e to not be able to distinguish between multiple values of the confounding variable z (discovered by the method in the previous section).

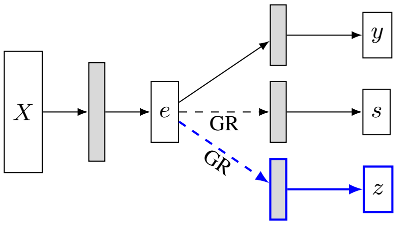

Figure 2: Adversarial classifier.

Similarly to the way we trained our topic model, we can use the training and testing datasets to fit this adversarial model. To do so, we take the losses from all the branches into account when fitting on instances from D_t and we only use the s-branch loss when fitting on instances from D_{t+}.

4.3 Proposed Approach and Baselines. In this section, we summarize the multiple models used to run our experiments. The simplest baseline we use is a logistic regression classifier (referred to as **LR**) that does not control for any kind of data shift. The remaining models are all different flavors of the architecture introduced in Figure 2. In Table 2, we indicate which branches of the architecture are active.

Using the same base architecture for most of our baselines makes the comparison easier as they all have the same number of hidden layers and the same number of neurons in each layer. The variation from one model to another stands in which branch is active and which is not.

- **DNN** stands for Dense Neural Network and is an non-adversarial approach, we obtain it by keeping the trunk and the highest branch of our shared model. Therefore, there are no gradient reversal layer in this model. It is expected to perform comparable to **LR**.
- The **DA** approach is the domain-adversarial approach developed by Ganin et al. [6] where no confounding variable information is available to the model.

	y-**branch**	s-**branch**	z-**branch**
DNN	✓	✗	✗
DA	✓	✓	✗
Ẑ	✓	✗	$\hat{z}$
DA+Ẑ	✓	✓	$\hat{z}$
Oracle	✓	✗	z

Table 2: Differences between models based on network architecture of Figure 2

- Model $\hat{\mathbf{Z}}$ is the first approach we propose in this paper: we first build an estimate of the latent confounding variable using the method described in this section and then we fit the adversarial model without the domain branch. We use the confidence σ_i to weight the y-loss for every training instance $\vec{x}_i$.

- Our second approach (**DA+$\hat{\mathbf{Z}}$**) combines models $\hat{\mathbf{Z}}$ and **DA** such that all branches are in use.

- Finally, the **Oracle** model is the same model as $\hat{\mathbf{Z}}$ with the crucial difference that the true confounding variable is known at training time: it provides a performance upper bound for the rest of the models. We are able to implement this model in the experiments below because we consider a setting in which we inject confounding shift into a dataset.

5 Datasets

We conducted experiments on three real-world datasets, in which the relationship between the confounding variable Z and the target variable Y changes between training and testing. For all experiments, we decide upon the Z variables ourselves, but none of the methods observe that variable directly (except for **Oracle**). This gives us an experimental framework to measure how robust the methods are to a known magnitude of confounding shift.

5.1 Twitter Datasets. This dataset contains Twitter users annotated with gender and location information. We create two classification tasks from this data: the first predicts location, where gender is a potential confound; the second predicts gender, where location is a potential confound. To build this dataset, we use Twitter streaming API to collect tweets with geo-coordinates from New York City (NYC) and Los Angeles (LA). Over a four-day period, we gathered a total of 246,930 tweets for NYC and 218,945 for LA. Afterwards, we further denoise our data by filtering out bots, celebrities, and marketing accounts by removing users with fewer than 10 followers or friends, more than 1,000 followers or friends, or more than 5,000 tweets. Using U.S. census name data, we then label unique users with their gender, removing any ambiguous names. For each user labeled, we collected all available tweets (up to 3,200) and represented each user as a binary unigram vector. Finally, we subsample from this collection and keep the tweets from 6,000 users such that gender and location are uniformly distributed over the users.

We refer to this as the **TwYLZG** dataset when $y_i = 1$ indicates NYC and $z_i = 1$ indicates Male. We also create the complementary **TwYGZL** dataset when $y_i = 1$ indicates Male and $z_i = 1$ indicates NYC. In this case, the task is to predict the user's gender, confounded by their location.

5.2 Yelp Dataset. We use the data provided through the Yelp Dataset Challenge to create this dataset. For each review in the dataset, we collect the text, the rating, as well as the category of the business targeted by the review. We encode the text using a binary vector to indicate the presence or absence of a given word. We then binarize the rating value such that reviews with ratings of 1 and 2 are assigned label 0 and those with ratings 4 and 5 are assigned label 1; neutral reviews (3-star rating) are removed from the dataset. Finally, we group businesses from food-related categories together (restaurants, bar, etc) and we group the remaining businesses together. This binary category encoding will be used in our experiments as the confounding variable such that $y_i = 1$ indicates a positive review and $z_i = 1$ indicates that the business targeted by the review is food-related. This dataset – referred to as **Yelp** – contains 6000 instances balanced across both y and z.

5.3 Injecting Confounding Shift. For the datasets we are working with, we know the true confounding variable. Thus, we can create datasets with different $P(Y|Z)$ distributions by sampling without replacement from each set $D_t^{'} \subseteq D_t$ and $D_{t+}^{'} \subseteq D_{t+}$ where y_i and z_i are binary variables. To simulate a change in $P(Y|Z)$, we introduce a bias parameter b where $P(y = 1|z = 1) = b$ and we use different bias terms b_t and b_{t+}, for training and testing. Thus, sampling was made according to the following constraints:

- $P_t(y = 1|z = 1) = b_t$
- $P_{t+}(y = 1|z = 1) = b_{t+}$
- $P_t(Y) = P_{t+}(Y)$
- $P_t(Z) = P_{t+}(Z)$

We emphasize the fact that we do not change any of the actual labels in the data, but instead merely sample instances to meet these constraints. While the first two are required to simulate a change in $P(Y|Z)$, the last two constraints are to isolate the effect of changes to $P(Y|Z)$; that is, we vary $P(Y|Z)$, but make $P(Y)$ and $P(Z)$ consistent from training to testing.

6 Experiments and Results

We first inject high confounding bias in datasets using the sampling method described above with biases $\{b_t, b_{t+}\} \in \{0.1, 0.9\}^2$. We create two datasets with every bias value such that we can repeat experiments. We fit our extended ProdLDA model with an Adam optimizer and a learning rate of 0.001 for at most 600 epochs. We set the dimensions of hidden layer to 256 and the topic layer contains 128 units. We train the proposed adversarial classifier with similar settings with a hidden layer of dimension 100 in the trunk part and hidden layers of dimension 50 in the branches parts.

	TwYLZG	TwYGZL	Yelp
LR	0.653 ± 0.02	0.694 ± 0.03	0.669 ± 0.03
DNN	0.650 ± 0.01	0.704 ± 0.02	0.655 ± 0.04
DA	0.809 ± 0.05	$\mathbf{0.779 \pm 0.04}$	$\mathbf{0.727 \pm 0.02}$
Ẑ	0.728 ± 0.11	0.641 ± 0.05	0.702 ± 0.04
DA+Ẑ	$\mathbf{0.848 \pm 0.01}$	0.773 ± 0.04	0.725 ± 0.04
Oracle	0.864 ± 0.01	0.823 ± 0.01	0.827 ± 0.02

Table 3: F1 results for the three tasks.

6.1 Main Results. Table 3 displays the F1 score for each of the methods across the three different tasks. The first thing to notice is that the model only focusing on domain adaptation (**DA**) is performing consistently well across three datasets and is robust to confounding shift. This makes sense as the confounding variable z and the data source s are highly correlated in case of high confounding shift.

Comparison with Generic Adversarial Domain Adaptation. The results indicate that our approach, **DA+Ẑ**, outperforms the generic adversarial approach **DA** on the first task by 3.9 points absolute improvement in F1. On the other tasks, its performance is on par with the generic adversarial approach.

Comparison with No Domain Adaptation. For all tasks, our approach outperforms a logistic regression classifier that performs no domain adaptation (**LR**), ranging from 5.6 to 19.5 absolute improvement points in F1. Our approach also outperforms **DNN** by 19.8, 7.9, and 7 points. It's also worth noting how severe high confounding shift is to baselines that ignore the confounds like **LR** and **DNN**. For example, for **TwYLZG** dataset, **DNN** and **LR** F1 scores degrade from 0.897 and 0.9 to 0.650 and 0.653, respectively, when high confounding bias is introduced into the dataset.

Comparison with an Oracle Approach. We are also comparing with an **Oracle** approach, where we assume we know the confounding variable ahead of time, and control for it directly using adversarial learning. As expected, **Oracle** outperforms all methods – it is worth noting that adversarial training does a very effective job of removing the influence of the confounder even under the extreme levels of confounding shift present here. For example, when there is no confounding shift, the **DNN** classifier on **TwYLZG** achieves 0.877 F1; under extreme confounding shift, this only drops to 0.864 F1 using adversarial training to control for the confound. Considering **Oracle** as a soft upper-bound, we can see that **DA+Ẑ** achieves most of the possible improvement over **DNN** that **Oracle** does. However, this is not the case on the other two datasets. We speculate that this is primarily due to the errors in identifying the likely confounding variable, as well as the attenuation bias produced by controlling for a noisy variable.

6.2 Additional Results In this section, we first analyze what makes the approach we propose outperform domain adaptation on **TwYLZG**. Then, we measure the sensitivity of the **DA+Ẑ** when the loss weight of z is varying.

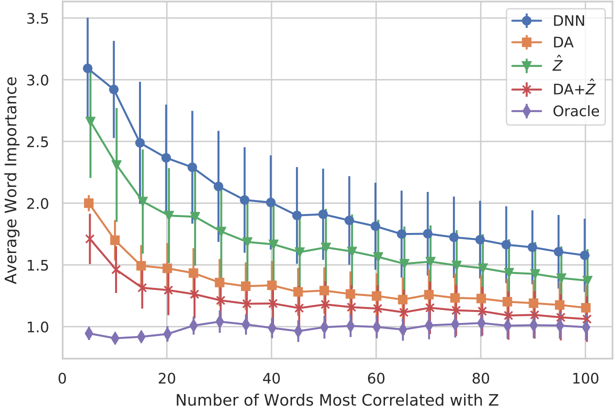

Figure 3: Average weight of words most correlated with the confounding variable.

Interpreting the Learned Representations. To better understand the difference between our approaches and the baselines, we analyzed the representations learned by each method, looking at how effective each approach was at removing features related to the confound from the model. Using the **TwYLZG** dataset, we first compute the words that have the strongest correlation with the gender label (e.g. "cute", "makeup", "lebron", etc.) when there is no confounding bias induced. Then, we train our model on biased train and test datasets displaying high confounding shift as in our previous experiments. Finally, we report the importance of the words most correlated with gender for each model. To do so, we compute the dot product from the input layer to the y-layer and retrieve the weight associated to every word. In order to compare different models, we standardize these weights and take their absolute value as we are only interested in the importance of the word but not the direction. Figure 3 shows the average weight of the K terms most correlated with gender with K varying from 5 to 100 on the x-axis. We observe a clear order where **DNN** puts a lot of importance on terms correlated with gender and then **DA**, **Ẑ**, **DA+Ẑ**, and **Oracle** each give less and less importance to these terms. In other words, each model controls more for words correlated with gender than the previous one, making it more robust in presence of confounding shift. This figure gives insight into why **DA+$\hat{\mathbf{Z}}$** outperforms **DA** on this dataset – the adversarial loss for $\hat{z}$ provides a greater penalty on representations that contain the confounding variable.

Additionally, Table 4 displays the most downweighted words when comparing **DNN** to **DA+$\hat{\mathbf{Z}}$**. As hinted by the results of Figure 3, these words are highly correlated with gender (e.g. nails, bro, boyfriend, causewereguys, etc). This shows that our approach is more robust to confounding shift by reducing the importance of words highly correlated with the confounding variable.

Sensitivity of DA+$\hat{\mathbf{Z}}$ to the Loss Weight on z. When controlling for a confounding variable using **DA+$\hat{\mathbf{Z}}$**, we can adjust the weight of the loss created by the z-branch. By default, the weights are set to 1 for all three losses in **DA+$\hat{\mathbf{Z}}$**. We can choose to favor the z-loss by increasing its weight or conversely we can favor the other two losses by decreasing z-loss weight. We make this weight vary from 0.001 to 100 when running experiments similar to our main experiments. We report the F1 scores of **DA+$\hat{\mathbf{Z}}$** on **TwYLZG** in Figure 4. For these experiments, keeping the three loss weights equal to 1 looks to be best method. Indeed, when z-loss weight is smaller than 1, the final F1 decreases by 3 to 4 points. Similarly, when this weight is greater than 1, the final F1 score drops by 2 points. Finally, it is important to note that not only the average F1 declines when changing the weight on z-loss but the model also becomes less stable as demonstrated by the larger error bars.

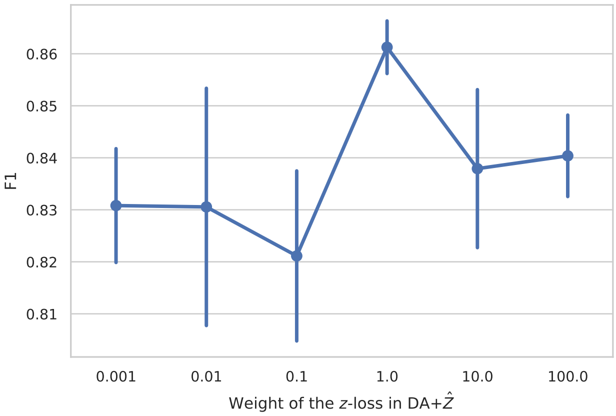

Figure 4: Performance of **DA+$\hat{\mathbf{Z}}$** when the weight on z-loss is varying from 0.001 to 100.

7 Limitations

Our approach seems to improve **DA** and gets closer to the **Oracle** model when the confounding variable is well captured by the topic model. Gender can be well captured through topics related to sports, video games, celebrities, or shopping, yielding high performance of our method in **TwYLZG**. However, location is much more difficult to model through topics created by ProdLDA which can explain why our approach is not improving over **DA** in the **TwYGZL** dataset. For the **Yelp** dataset, we hypothesize that the confounding variable food vs other business can be captured by ProdLDA but is diluted across many topics. Indeed, food can be broken down in a lot of subtopics (e.g. mexican, japanese, french, etc) and other businesses range from car dealers to nail salons, making the approach less effective. Although this is a good first step to bridge the gap between domain adaptation and confounding shift control, we wish to address this issue in future work and to provide a topic model that would capture these confounding variables.

8 Conclusion

In this paper, we presented a two step approach to control for confounding shift caused by an unknown confounding variable. We first use an adversarial topic model to discover words that are changing from training time to testing time. We then use these words to assign an estimated confounding variable value to every document in the dataset, and we control for this using an adversarial training objective. We show on three datasets that domain adaptation manages to recover well from confounding shift although it is not its primary purpose. Additionally, our results show that our proposed approach is able to bridge part of the gap towards an **Oracle** method aware of the confounding variable when the latent confounding variable is well captured by the topic model.

Acknowledgments

This research was funded in part by the National Science Foundation under awards #IIS-1526674 and #IIS-1618244.

References

[1] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of machine Learning Research*, 2003.

[2] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128. Association for Computational Linguistics, 2006.

[3] Xilun Chen and Claire Cardie. Multinomial adversarial networks for multi-domain text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, volume 1, pages 1226–1240, 2018.

[4] Harrison Edwards and Amos Storkey. Censoring representations with an adversary. In *ICLR*, 2016.

nails cute makeup yay wonderful adorable bottle bro omg husband sir excited hubby xoxo 3rd pls cried understands
frozen boyfriend waiting seth bros 300 la slip idiot case boo daughter fingers main heels bought experience roomie
records packing fellas necessary happiest baking haters flag pinterest showing causewereguys oscar starter announce
anxiety heart ladies champagne ski noodle midnight sunshine bff asian closet gossip lovely eyebrows 16 pace cousin
loved metal broken ppl princess nyc tweetlikeagiri draft seems laughing bird congratulations until followers bust woman
marcus luke shops goodness female eats smoking chill round map perspective women chocolate line denial fam pink

Table 4: 100 most downweighted words when moving from **DNN** to **DA+$\hat{\mathbf{Z}}$**

[5] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):44, 2014.

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[8] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[9] Virgile Landeiro and Aron Culotta. Robust text classification in the presence of confounding bias. In *Thirtieth National Conference on Artificial Intelligence (AAAI)*, 2016.

[10] Patrick Lindstrom, Sarah Jane Delany, and Brian Mac Namee. Handling concept drift in a text data stream constrained by high labelling cost. In *FLAIRS Conference*, 2010.

[11] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. In *ICML*, 2018.

[12] Alexandra Olteanu, Onur Varol, and Emre Kiciman. Distilling the outcomes of personal experiences: A propensity-scored analysis of social media. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, pages 370–386. ACM, 2017.

[13] Reid Pryzant, Kelly Shen, Dan Jurafsky, and Stefan Wagner. Deconfounded lexicon induction for interpretable social science. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1615–1625, 2018.

[14] Edward Raff and Jared Sylvester. Gradient reversal against discrimination. In *Proceedings of the 5th Workshop on Fairness, Accountability and Transparency in Machine Learning*, 2018.

[15] Akash Srivastava and Charles Sutton. Autoencoding variational inference for topic models. In *ICLR*, 2017.

[16] Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. Controllable invariance through adversarial feature learning. In *Advances in Neural Information Processing Systems*, pages 585–596, 2017.

[17] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *AAAI*, 2018.