FEUDA: Frustratingly Easy Prompt Based Unsupervised Domain Adaptation

Anonymous ACL submission

Abstract

A major thread of unsupervised domain adaptation (UDA) methods uses unlabeled data from both source and target domains to learn domain-invariant representations for adaptation. However, these methods showcase certain limitations, encouraging the use of self-supervised learning through continued pre-training. The necessity of continued pre-training or learning domain-invariant representations is still unclear in the prompt-based classification framework, where an input example is modified by a template and then fed into a language model (LM) to generate a label string. To examine this new paradigm of UDA in the prompt-based setup, we propose a frustratingly easy UDA method (FEUDA) that trains an autoregressive LM on both unlabeled and labeled examples using two different instruction-tuning tasks. Specifically, the first task trains the LM on unlabeled texts from both domains via masked language modeling (MLM), and the other uses supervised instruction-tuning on source-labeled data for classification. We conduct extensive experiments on 24 real-world domain pairs to show the effectiveness of our method over strong domain-invariant learning methods. Our analysis sheds light on why masked language modeling improves target-domain classification performance in prompt-based UDA. We discover that MLM helps the model learn both semantic and background knowledge of a domain, which are both beneficial for downstream classification.

1 Introduction

Despite recent advancements in the pre-training of language models, these models are still fragile under certain kinds of data distribution shifts, masking their real-world applications challenging (Ribeiro et al., 2020). The problem of unsupervised domain adaptation (UDA) aims to leverage learned knowledge from a labeled source domain to an unlabeled target domain (Pan and Yang, 2010; Ganin and Lempitsky, 2015; Long et al., 2015).

A vast class of existing UDA methods attempts to learn representations that are invariant across domains (Tzeng et al., 2014; Ganin et al., 2016; Wu and Shi, 2022; Guo et al., 2022). The rationale is that when the learned representations from both domains cannot be distinguished by a classifier and the classifier performs well on the source domain, it will also exhibit strong performance on the target domain. However, previous work has shown that domain-invariance is insufficient for adaptation to the target domain (Zhao et al., 2019), and is also prone to instability issues (Han and Eisenstein, 2019; Kashyap et al., 2021). This has encouraged the emergence of self-supervised approaches through language model pre-training. Variants of continued pre-training have proven to be effective and stable for adapting pre-trained LMs to labeled and unlabeled downstream tasks (Gururangan et al., 2020; Karouzos et al., 2021).

However, the trade-offs between continued pre-training and learning domain-invariant representations for UDA are both unexplored in the prompt-based classification framework (Gao et al., 2021; Liu et al., 2023). In such a framework, input examples are modified using instruction templates and then fed into a language model (LM) to generate the label text based on the constructed instruction. This process bears resemblance to instruction tuning (Wei et al., 2022), except that the training is done on a single task, and that the end goal is adaptation to a specific task, rather than generalization to unseen tasks. In this paper, we call this new paradigm *prompt-based UDA* and examine two research questions: *Can we utilize unlabeled data to construct useful instruction-tuning tasks for UDA? Is domain invariance still necessary in this paradigm?* To answer these questions, we present a "frustratingly easy" UDA method (termed **FEUDA**), which smooths out the transition between pre-training and adaptation by two instruction-tuning tasks using prompt templates.

First, unlabeled texts from both source and target domains are modified by a prompt template and then used to train an autoregressive LM to perform the masked language modeling (MLM) task. Next, the LM is instruction-tuned on labeled source texts using another template for the classification task.

Through extensive experiments on 40 real-world domain pairs, various adaptation methods, and few-shot learning setups, we show that FEUDA is competitive for UDA and even outperforms methods that explicitly promote domain invariance (Section 6.1). Additionally, our analysis sheds light on how masked language modeling improves classification performance on the unlabeled target domain (Section 6.2). We discover that MLM helps the model learn both semantic and background knowledge of a domain, which are both beneficial for downstream classification. Our main contributions can be summarized as follows:

1. We introduce prompt-based UDA, a new UDA setting where the discriminative prediction is converted into a generative task, enabling multi-task adaptation as well as the reuse of all language model parameters. We empirically analyze continued pre-training and domain invariance based UDA methods in this setting.
2. We propose FEUDA, a simple and effective UDA approach for prompt-based classification. Through extensive experiments, we show that FEUDA is competitive and outperforms the domain-invariant learning approach. We establish the generalizability of FEUDA across various models, adaption methods and limited data settings, confirming that our approach remains powerful in these settings.
3. We conduct an analysis understanding the impact of the MLM task in a UDA setup and discover that MLM helps the model learn both semantic and background knowledge of a domain, both of which are beneficial for downstream classification.

2 Related Work

Ramponi and Plank (2020) categorize UDA methods into two general classes: Model-centric and Data-centric methods. This work focuses on a new data-centric UDA method in a prompt-based setup.

Model-centric UDA Methods This line of study involves augmenting the feature space (Blitzer et al., 2006; Pan et al., 2010; Ziser and Reichart, 2018, *inter alia*), editing models though weight interpolation (Matena and Raffel, 2022; Cai et al., 2023; Wortsman et al., 2022; Ilharco et al., 2022), or altering the loss function and model architecture. One typical framework aims to minimize $\mathcal{H}\Delta\mathcal{H}$ divergence (Ben-David et al., 2010) between the source and target domain features, through adversarial training (Tzeng et al., 2014; Ganin et al., 2016; Tzeng et al., 2017; Wu and Shi, 2022; Guo et al., 2022, *inter alia*) or through minimizing measures of domain similarity (Bousmalis et al., 2016; Ge et al., 2023; Malik et al., 2023). However, past work has shown that domain-invariance is weak constraint for adaptation to the target domain (Zhao et al., 2019; Karouzos et al., 2021), could introduce domain-specific hyperparameters (Trung et al., 2022), and is also prone to instability issues (Han and Eisenstein, 2019; Sun et al., 2019; Wilson and Cook, 2020; Kashyap et al., 2021).

Data-centric UDA Methods The limitations of invariance-based model-centric methods have encouraged the emergence of alternate approaches, largely based on self-supervised learning through contrastive learning (Kumar et al., 2022; Shen et al., 2022; Long et al., 2022), pseudo-labeling (Zhou and Li, 2005; Ruder and Plank, 2017, *inter alia*) or language model pre-training. Despite not being directly useful to certain downstream tasks (Uppaal et al., 2023), Masked Language Modelling (MLM) has been used for adaptation to labeled tasks, in both full fine-tuning (Gururangan et al., 2020; Lee et al., 2020; Gao et al., 2021) and PEFT setups (Kim et al., 2021; Hung et al., 2023). A smaller body of work has explored the utility of MLM in a UDA setup (Han and Eisenstein, 2019; Zhang et al., 2021b; Karouzos et al., 2021). This class of methods is more stable than invariance-based methods, but often requires additional compute for extended pre-training.

Prompt-based UDA The emergence of large language models (Brown et al., 2020; Scao et al., 2022; Touvron et al., 2023, *inter alia*) introduced the concept of instruction tuning, where a language model is trained on strings of input-output pairs, often using instruction-specific templates (Zhang et al., 2023). Inspired by this, we introduce prompt-based UDA, i.e., a new paradigm of data-centric UDA using a prompt-based classifier that casts the discriminative classification task into a generative next-token prediction task. This prompt-based UDA formulation provides two unique benefits compared to traditional UDA approaches: 1) the prompt-based classifier can reuse all model parameters for adaptation, without requiring any task-specific architectural changes; 2) furthermore, this also enables multi-task instruction tuning bridging the gap between pre-training and adaptation, as the pre-training and fine-tuning phases of the exact model can be naturally coupled together by using different instruction prompts.

Our study aims to extend the existing body of data-centric UDA methods by examining the behavior of multi-task instruction tuning for adaptation and the necessity of learning domain invariant representations in this new UDA paradigm.

3 Preliminaries: The UDA Problem

We consider a text classification task, where $\mathcal{X}$ is the input space of all text sentences and $\mathcal{Y} = \{1, ...K\}$ is the label space. In the UDA problem, we have access to a source labeled dataset $\mathcal{D}_{\text{src}} = \{(x_i, y_i)\}_{i=1}^N$ consisting of samples from a joint distribution $P_{\mathcal{XY}}^{(\text{src})}$, and a target unlabeled dataset $\mathcal{D}_{\text{tgt}} = \{x_j\}_{j=1}^M$ sampling from a target input distribution $P_{\mathcal{X}}^{(\text{tgt})}$. We further denote $P_{\mathcal{X}}^{(\text{src})}$ as the marginal distribution of $P_{\mathcal{XY}}^{(\text{src})}$ on $\mathcal{X}$, where $P_{\mathcal{X}}^{(\text{src})} \neq P_{\mathcal{X}}^{(\text{tgt})}$. The goal of UDA is to learn a function $f : \mathcal{X} \to \mathcal{Y}$ such that the error rate $\mathbb{E}_{x \sim P_{\mathcal{X}}^{(\text{tgt})}} \mathbb{1}[f(x) \neq y]$ is minimized.

4 FEUDA Method

In this section, we introduce our framework FEUDA, a simple and effective two-phase training method[1] for UDA with masked language modeling. In the first phase, we take a pre-trained autoregressive language model and perform unsupervised training through masked language modeling, on the combination of unlabeled data from both the source and target domains. In the second phase, we perform supervised fine-tuning on the downstream classification task using labeled data from the source domain.

Task 1: MLM Pre-training We aim to utilize a pre-trained autoregressive language model for continuing pre-training on unlabeled data from both the source and target domains, with the masked language modeling task. Here, we reuse all the input sequences from the source-labeled dataset $\mathcal{D}_{\text{src}}$ as the source-unlabeled dataset, denoted as $\mathcal{D}_{\text{src}}^x$. Next, similar to (Raffel et al., 2020), for any unlabeled sequence $x \in \mathcal{D}_{\text{src}}^x$ and $\mathcal{D}_{\text{src}}$, we use a prompt template to convert the sequence x to an input-output sequence pair, i.e., $\mathbb{M}(x) = (\tilde{x}, \tilde{y})$. Here, the prompt template first randomly masks words in the input sequence x and prepends an instruction (i.e., "Fill in the blanks:") to create a new input sequence $\tilde{x}$. An output sequence $\tilde{y}$ is then constructed by concatenating all the masked words separated by a special <sep> token. For example,

$x =$ The movie was so cool! Two hours of fun.

$\tilde{x} =$ Fill in the blanks: "The _ cool! Two hours _

$\tilde{y} =$ <sep> movie was so <sep> of fun. <sep>

Given a pair $(\tilde{x}, \tilde{y})$, we use an autoregressive LM parameterized by θ to compute the negative log-likelihood loss averaged over output words:

$$\ell(\tilde{x}, \tilde{y}; \theta) = -\frac{1}{|\tilde{y}|} \sum_t \log P_\theta(\tilde{y}_t | \tilde{x}, \tilde{y}_{1:t-1}). \quad (1)$$

Notably, we convert the MLM task into a next-token prediction task by instructing an autoregressive LM to predict the output words, which allows us to reuse all the parameters of the LM without adding any new randomly-initialized parameters. Finally, we define the total loss on the combined unlabeled dataset $\mathcal{D} = \mathcal{D}_{\text{src}}^x \cup \mathcal{D}_{\text{tgt}}$ as:

$$\mathcal{L}_{\text{MLM}}(\mathcal{D}; \theta) = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \ell(\mathbb{M}(x); \theta). \quad (2)$$

While the above formulation uses the MLM task for continued pre-training, the use of an autoregressive LM allows for an easy extension to the Causal Language Modeling (CLM) task, which we demonstrate in Section 6.3.

Task 2: Source Supervised Instruction-tuning In the second phase, we only use labeled data from the source domain to fine-tune the model for the downstream classification task. Similar to the first phase, we use a prompt template[2] to create input-output sequence pairs from the labeled data in the source domain. Specifically, for a labeled example $(x, y) \in \mathcal{D}_{\text{src}}$, a new prompt template appends to x an instruction for prompting the LM to perform classification, and converts the label y to its corresponding text description $\tilde{y}$, i.e., $\mathbb{C}(x, y) = (\tilde{x}, \tilde{y})$. For example, a labeled example with the positive

[1]While we use the two-phase multi-task training pipeline (sequential) in our main experiments, in Appendix B, we show that an equivalent single-phase multi-task training pipeline (joint) results in similar performance.

[2]Prompt templates were selected from the Public Pool of Prompts (Bach et al., 2022).

sentiment from a sentiment classification task is converted as follows:

$x =$ I like this movie. $\quad y = 1$

$\tilde{x} = [x]$ Is this sentence positive or negative?

$\tilde{y} =$ Positive

Given the augmented sequence pair $(\tilde{x}, \tilde{y})$ and the model trained after the first phase, we compute the same negative log-likelihood loss $\ell(\tilde{x}, \tilde{y}; \theta)$ in Eq. (1). This again allows us to reuse all model parameters, including the language model prediction head. Finally, we define the total loss on the source-labeled dataset in the second phase as:

$$\mathcal{L}_{\text{CLS}}(\mathcal{D}_{\text{src}}; \theta) = \frac{1}{|\mathcal{D}_{\text{src}}|} \sum_{(x,y)\in\mathcal{D}_{\text{src}}} l(\mathbb{C}(x, y); \theta). \tag{3}$$

After training, we follow the practice of Liu et al. (2022) to convert a label string $\tilde{y}$ to its corresponding label y at test time for evaluation.

Parameter-Efficient Fine-Tuning (PEFT) The above formulation is general and can be applied to fine-tune all model parameters. Additionally, our FEUDA framework is compatible with the parameter-efficient fine-tuning approach. The PEFT approach is desirable because it adds only a small amount of learnable parameters ϕ to a pre-trained language model θ, and fine-tunes only ϕ to perform prediction while keeping the other model parameters θ frozen. We utilize two instantiations in our implementations: Adapters (Houlsby et al., 2019) and (IA)^3 (Liu et al., 2022). Both are high-performing PEFT approaches, with (IA)^3 using fewer learnable parameters. More details about both methods can be found in Appendix A.

5 Experimental Setup

5.1 Datasets

We follow the setup from Malik et al. (2023), and use two sentence classification datasets with 5 domains each. This results in a total of 40 pairs of source and target domains. Appendix C shows more details about the evaluation benchmarks.

MNLI The MNLI corpus (Williams et al., 2018) contains sentence pairs across multiple genres: Travel (T), Fiction (F), Government (G), Slate (S), and Telephone (Te). The task classifies every sentence pair as entailment, neutral, or contradiction.

Amazon The Multi-Domain Sentiment Analysis Dataset (Blitzer et al., 2007) contains Amazon product reviews for different types of products. We use reviews from the Apparel (A), Baby (B), Books (Bo), Cameras (C), and Movies (M) domains. Each review is labeled as either positive or negative.

5.2 Models and Training

Models For our main experiments, we use T5v1.1, which is an improved version of the original T5 model (Raffel et al., 2020). Unlike the original T5 model, T5v1.1 is not trained on any supervised datasets. We use the base (60M parameters) and XL (3B parameters) versions of the model. We also use T0 (3B parameters), which has been optimized for zero-shot generalization by training on supervised natural language prompts (Sanh et al., 2022). Furthermore, to test the sensitivity of FEUDA across architectures and masking styles, we also use GPT-2 medium (345M parameters) (Radford et al., 2019), from the class of autoregressive decoder-only language models.

Training We train each training phase for 30,000 steps on MNLI and 15,000 steps on the Amazon dataset. We train with Adam and use a batch size of 8, learning rate of 0.003. We set the maximum sequence length to 256 tokens. We use length normalization during evaluation, as proposed by Liu et al. (2022). For each experiment, we report the mean and standard deviation across 3 runs. More details can be found in Appendix D.

5.3 Baselines

We compare **FEUDA** with three baselines below.

- **Src Only**: We fine-tune the model on the source labeled data, in a single training phase.
- **Src+Tgt** (All labeled): We fine-tune the model for classification, using labeled data from both the source and target domains. This serves as an upper bound on target domain performance.
- **MMD**: The current state of the art for UDA in a PEFT setup promotes domain invariance by maximum mean discrepancy (MMD) (Malik et al., 2023). This method measures the multi-kernel maximum mean discrepancy (MK-MMD) (Gretton et al., 2012; Bousmalis et al., 2016) between source and target embeddings from each transformer layer and sums them to obtain an aggregate loss $\mathcal{L}_{\text{div}}$. The final loss is the weighted sum of $\mathcal{L}_{\text{div}}$ and the classification loss, i.e., $\mathcal{L} = \lambda\, \mathcal{L}_{\text{cls}} + (1 - \lambda)\, \mathcal{L}_{\text{div}}$, where λ gradually changes from 0 to 1 during training.

Source → Target	Source Accuracy				Target Accuracy			
	Src Only	Src+Tgt	MMD	FEUDA	Src Only	Src+Tgt	MMD	FEUDA
T->F	78.4 (2.2)	79.1 (0.5)	78.3 (0.1)	**80.5** (0.1)	73.7 (1.1)	77.2 (0.4)	69.7 (0.8)	**74.1** (0.9)
T->G	78.4 (2.2)	79.8 (0.1)	78.2 (0.2)	**80.1** (0.3)	73.7 (1.1)	83.6 (0.7)	79.3 (0.5)	**83.6** (0.3)
T->S	78.4 (2.2)	80.6 (3.5)	**79.8** (0.2)	**79.8** (0.4)	73.7 (1.1)	72.3 (0.5)	69.6 (0.1)	**70.7** (0.6)
T->Te	78.4 (2.2)	79.2 (0.4)	78.0 (0.0)	**81.1** (0.0)	74.5 (2.2)	77.8 (0.1)	69.4 (0.8)	**76.8** (0.0)
F->T	76.0 (0.2)	77.6 (0.3)	**72.9** (0.2)	67.6 (1.5)	75.6 (0.7)	79.9 (0.1)	**69.9** (0.2)	65.4 (1.8)
F->G	76.0 (0.2)	77.6 (0.6)	53.3 (21.4)	**73.2** (2.3)	75.6 (0.7)	82.3 (0.1)	54.3 (23.4)	**78.8** (2.5)
F->S	76.0 (0.2)	77.4 (0.3)	69.7 (2.7)	**69.8** (1.8)	75.6 (0.7)	72.1 (0.2)	64.6 (1.8)	**65.3** (1.6)
F->Te	76.0 (0.2)	77.8 (0.6)	70.6 (0.9)	**74.4** (0.4)	75.6 (0.7)	78.3 (0.6)	64.6 (0.7)	**72.5** (0.2)
G->T	82.1 (0.3)	83.6 (0.4)	80.9 (0.7)	**82.3** (0.8)	73.0 (0.0)	79.9 (0.4)	**75.9** (0.3)	75.8 (0.6)
G->F	82.1 (0.3)	81.6 (0.1)	79.8 (0.7)	**81.7** (0.2)	73.0 (0.0)	76.7 (0.1)	69.9 (0.2)	**73.5** (0.2)
G->S	82.1 (0.3)	82.9 (0.1)	**80.9** (0.0)	79.8 (1.7)	73.0 (0.0)	73.1 (0.0)	**69.4** (0.1)	68.0 (1.8)
G->Te	82.1 (0.3)	83.2 (0.2)	80.1 (0.1)	**82.1** (0.1)	73.0 (0.0)	78.1 (0.6)	69.9 (0.3)	**73.5** (0.6)
S->T	70.9 (1.7)	71.9 (0.1)	69.1 (0.9)	**71.2** (0.2)	72.9 (1.5)	79.5 (0.3)	74.4 (1.7)	**76.8** (1.0)
S->F	70.9 (1.7)	71.6 (0.4)	**70.4** (0.0)	70.3 (0.7)	72.9 (1.5)	77.7 (0.2)	**73.1** (0.0)	72.4 (0.7)
S->G	70.9 (1.7)	72.8 (0.2)	**68.5** (0.8)	66.4 (1.5)	72.9 (1.5)	83.4 (0.2)	**78.2** (0.5)	76.3 (0.9)
S->Te	70.9 (1.7)	73.3 (0.0)	67.5 (1.4)	**71.0** (1.7)	72.9 (1.5)	78.5 (0.0)	66.7 (0.2)	**74.8** (1.3)
Te->T	77.5 (0.2)	78.2 (0.4)	75.5 (0.5)	**78.7** (0.2)	74.9 (0.2)	79.8 (0.3)	71.4 (0.0)	**76.5** (0.4)
Te->F	77.5 (0.2)	78.1 (0.4)	75.2 (0.7)	**77.1** (0.0)	74.9 (0.2)	77.9 (0.1)	69.9 (0.5)	**74.3** (0.5)
Te->G	77.5 (0.2)	78.6 (0.2)	74.8 (0.5)	**78.8** (0.1)	74.9 (0.2)	82.5 (0.1)	75.6 (1.6)	**82.0** (0.6)
Te->S	77.5 (0.2)	78.7 (0.0)	75.3 (0.1)	**78.8** (0.4)	74.9 (0.2)	72.2 (0.0)	68.0 (0.4)	**71.3** (0.5)

Table 1: Comparison of FEUDA and MMD by classification accuracy on the MNLI dataset, using the T5v1.1 base model, and (IA)^3 PEFT method. FEUDA is competitive with MMD, often outperforming it. The highest values between FEUDA and MMD have been marked in bold. Cases where Src Only outperforms both FEUDA and MMD on the target have been underlined. However, it must be noted that in a majority of these cases the upper bound Src+Tgt is comparable to or weaker than Src Only, indicating noise in the domain pair.

Source → Target	Source Accuracy				Target Accuracy			
	Src Only	Src+Tgt	MMD	FEUDA	Src Only	Src+Tgt	MMD	FEUDA
A->B	93.7 (0.1)	93.8 (0.3)	**94.3** (0.2)	93.1 (0.3)	93.3 (0.4)	94.7 (0.2)	93.8 (0.3)	**93.9** (0.3)
A->B	93.7 (0.1)	94.2 (0.1)	**93.8** (0.1)	92.8 (0.6)	90.8 (0.6)	94.3 (0.4)	**92.5** (1.1)	90.2 (1.2)
A->C	93.7 (0.1)	93.4 (0.3)	**95.0** (0.0)	93.9 (0.5)	91.9 (0.1)	95.0 (0.2)	91.8 (0.5)	**92.1** (0.5)
A->M	93.7 (0.1)	94.1 (0.3)	**94.7** (0.3)	93.5 (0.4)	81.3 (1.4)	85.8 (0.5)	81.3 (0.6)	**83.3** (0.5)

Table 2: Comparison of FEUDA and MMD by classification accuracy on the Amazon Product Review dataset, using the T5v1.1 base model, and (IA)^3 PEFT method. FEUDA is competitive with MMD, often outperforming it. The highest values between FEUDA and MMD on the target domain have been marked in bold.

6 Results & Analysis

6.1 FEUDA is Competitive for UDA

We compare our method with other baselines over 24 domain pairs (16 additional pairs in Appendix E) domain pairs across the MNLI and Amazon product review datasets. In these experiments, we use the T5v1.1 Base model and (IA)^3 PEFT method.

FEUDA outperforms methods that explicitly promote domain invariance Table 1 shows the classification accuracy on the MNLI dataset. We find that FEUDA is a competitive method to MMD. For example, for Travel (T) → Government (G), FEUDA yields an accuracy of 83.6% on the target domain, equalling the upper bound of the Src+Tgt baseline. In comparison, MMD yields an accuracy of only 79.3%. Additionally, FEUDA performs more stably than MMD. For example, for Fiction (F) → Government (G), minimizing MMD yields a variance of over 20% across runs. This observation is consistent with existing findings (Kashyap et al., 2021; Han and Eisenstein, 2019) that minimizing divergence measures like MMD, when combined with auxiliary task-specific loss functions, result in training instabilities and vanishing gradients. We discuss the performance of the MMD method in more detail in Appendix J.

We also see similar results on the Amazon dataset in Table 2 (full results in Appendix E). For example, for Apparel (A) → Movies (M), FEUDA yields an accuracy of 83.3% on the target domain, approaching the upper bound of the Src+Tgt baseline. In comparison, MMD yields an accuracy of 81.3%. A visualization of sentence embeddings in Figure 4 (Appendix E) suggests that representations learned through FEUDA are not domain invariant. In Appendix I, we show FEUDA outperforms additional baselines, including DANN (Ganin et al., 2016), weight interpolation (Ilharco et al., 2022), and domain divergence minimization by Wasserstein distance.

6.2 Analyzing the Impact of MLM on UDA

In this section, we aim to understand how MLM training on the source and target domains boosts classification on the unlabeled target domain.

Impact of Masked Words We hypothesize that by having to predict certain masked words during MLM training, the model implicitly learns information about the classification task on the unlabeled domain. For example, given the masked sentence,

"I really _ the movie, it was a fascinating watch."

The only way the model can predict the masked word is by using the sentence context and identifying the sentiment of certain words. In this case, the word "fascinating" implies a positive sentiment, so the model may predict the masked word to be a positive word like "loved" or "enjoyed". Thus, the model would implicitly learn information about the downstream task, by predicting masked words (e.g., "fascinating") indicative of the class label.

To test this hypothesis, we quantize the "informativeness" of each word to a classification task. An informative word is one that is highly correlated with any of the labels in the downstream task. Specifically, we follow Gururangan et al. (2018) and use pointwise mutual information (PMI) (Fano, 1961) of the word with respect to the class label:

$$\mathrm{PMI}(\mathrm{word}, \mathrm{class}) = \log \frac{p(\mathrm{word}, \mathrm{class})}{p(\mathrm{word})p(\mathrm{class})},$$

where we count the frequency of a word-class pair on the labeled data $\mathcal{D}_{\mathrm{src}}$ to estimate $p(\mathrm{word}, \mathrm{class})$, and similarly estimate $p(\mathrm{word})$ and $p(\mathrm{class})$ by counting a word and a class individually on $\mathcal{D}_{\mathrm{src}}$. These informative words are similar to pivot features (Blitzer et al., 2006; Ben-David et al., 2020, *inter alia*), with the exception that they are chosen based on information from the source domain only.

To compare with random masking of $k\%$ words, we selectively mask the top $k\%$ or bottom $k\%$ of informative words in a sentence, ranked by their PMI with any inference label ($k = 15$). We also filter out low-frequency words from the selection.[3] We use the T5v1.1 base model with (IA)[3] on the Apparel $\rightarrow$ Movies pair for analyzing the impact of masked words at *inference* and *pre-training* time.

Impact at Inference: We use a prompt-based classifier trained by FEUDA to classify three versions of a test set at inference: the original test set and the other two versions with informative and uninformative words masked respectively. Figure 1 (corresponding Table 10 in Appendix G) shows us that the presence of *both* informative and uninformative words are essential for strong classification performance, with performance being highest on the original unmasked sequences. Interestingly, the source-domain performance is only hurt by the masking of informative words, confirming that these words are highly indicative of the downstream classes.

Impact at Pre-training: To further confirm the phenomenon, we alter the masking strategies in the MLM pre-training phase of FEUDA. We compare the original random masking with informative and uninformative masking, maintaining a fixed masking rate (15%) across masking strategies. Table 3 confirms that random masking is most helpful for target-domain classification. To isolate any effects of PEFT methods or pre-training data, we repeat the analysis by fine-tuning the instruction-tuned Flan-T5 (Chung et al., 2022) and notice a similar trend (Table 11 in Appendix G). We hypothesize that the model learns semantic features through the masking of informative words and background features through the masking of uninformative words, and both sets of features are essential for classification on the unlabeled domain.

Masking Strategy	Accuracy Source	Accuracy Target
Random	**93.5** (0.4)	**83.3** (0.5)
Informative	93.3 (0.5)	78.3 (0.7)
Uninformative	92.9 (0.1)	79.6 (0.6)

Table 3: Impact of word selection for masking during training. Masking words at random is more powerful than selectively masking informative or uninformative words. This indicates that the model requires *both* semantic features (learnt through masking informative words) and background features (learnt through masking uninformative words) for classification on the unlabelled target domain.

[3]Any word that occurs less than 10 times in the entire training corpus is considered to be low frequency.

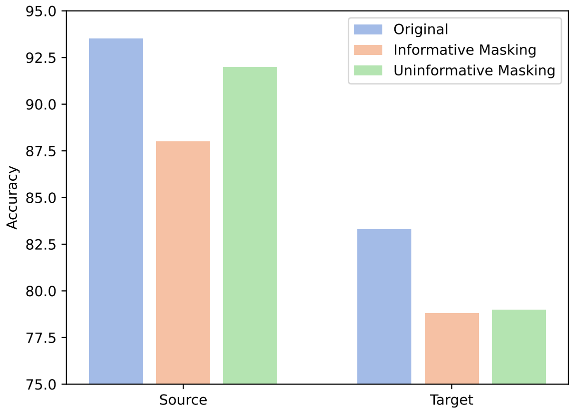

Figure 1: Impact of masking at inference. We evaluate FEUDA on the Apparel → Movies domain pair, and select words for masking based on their "informativeness" to the classification task. The performance of the model is best with the original unmasked sequences, indicating the presence of *both* informative and uninformative words are essential.

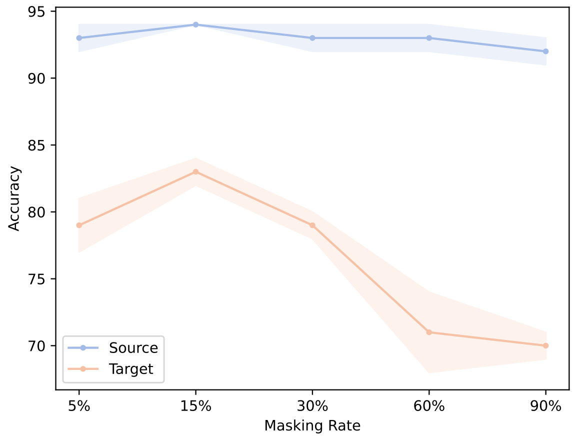

Figure 2: Impact of Masking Rate on FEUDA. With high masking rates, the performance on the source domain is largely maintained, but the performance on the target domain rapidly deteriorates.

Impact of Masking Rate While masking 15% of a sequence is considered standard for random masking, previous work has shown that BERT-sized models (Devlin et al., 2019) can learn from as high as 80% masking rates during pre-training followed by adaptation to a labeled task. Here, we explore the role of masking rates during continued pre-training for adaptation to an unlabeled task.

Figure 2 (Table 12 in Appendix) shows the impact of varying the masking rate on the source and target domains. With masking rates under the optimal value of 15%, the semantic and background features learned through model prediction of masked words is limited, hurting performance on the target domain. Beyond the 15% rate, the classification performance on the source domain is largely maintained, even at a 90% masking rate, matching previous findings (Wettig et al., 2023). However, the performance on the target domain rapidly decreases with an increasing masking rate.

To explain the performance drop, we hypothesize that since the model never sees any labeled data of the target domain, it heavily depends on the signal it gets from the unlabeled data through masking. Effectively masking a majority of a sequence removes the background and semantic features from the sequence, both of which are necessary for downstream classification on the domain.

6.3 Extensions to More Settings

Prompting is common practice with large language models which are too compute-intensive for traditional fine-tuning. For the same reasons, the setup is also frequently combined with learning from limited examples and parameter-efficient fine-tuning. In this section, we explore these settings, using the Apparel (A) → Movies (M) domain pair from the Amazon Reviews dataset.

Model Types & Scales We evaluate the performance of FEUDA over larger and instruction tuned encoder-decoder models, using T5v1.1 XL and T0 (Sanh et al., 2022). Table 9 (Appendix F) shows a wider gap between MMD and FEUDA with higher model capacity, and this gap is further increased with instruction tuning. We also explore the utility of FEUDA when using causal language modeling (CLM) with a decoder-only language model. Table 5 shows us that FEUDA provides strong improvements on both domains, equalling the Src+Tgt baseline on the target domain.

Adaptation Methods PEFT approaches have been shown to introduce resilience to domain shift (Fu et al., 2023). To isolate this effect from the FEUDA framework, we evaluate our method in a full-fine-tuning setup. Further, we compare with two PEFT approaches: Adapters (Houlsby et al., 2019) and $(IA)^3$ (Liu et al., 2022). We choose Adapters because He et al. (2022) present a unified view of PEFT approaches which shows that the operations applied by Adapters are very similar to those of Prefix Tuning (Li and Liang, 2021) and LoRA (Hu et al., 2022). We choose $(IA)^3$ since it is a state-of-the-art PEFT approach that uses a fraction of the learnable parameters of Adapters.

Method	Source Accuracy				Target Accuracy			
	Src Only	Src+Tgt	MMD	FEUDA	Src Only	Src+Tgt	MMD	FEUDA
Fine-Tuning	93.9 (0.5)	94.0 (0.4)	93.8 (0.3)	**95.0** (0.4)	82.0 (1.1)	86.4 (0.4)	82.4 (1.6)	**84.4** (0.3)
(IA)3	93.7 (0.1)	94.1 (0.3)	**94.7** (0.3)	93.5 (0.4)	81.3 (1.4)	85.8 (0.5)	81.3 (0.6)	**83.3** (0.5)
Adapters	93.6 (0.1)	94.6 (0.3)	**94.4** (0.7)	**94.3** (0.2)	80.8 (1.3)	85.3 (0.5)	79.1 (0.3)	**82.7** (0.5)

Table 4: Performance of FEUDA across different adaptation methods with the T5v1.1 base model on the Apparel $\to$ Movies domain pair. FEUDA remains more powerful than MMD across all methods.

Method	Accuracy	
	Source	Target
Src Only	86.9 (1.4)	65.9 (1.4)
Src+Tgt	86.8 (1.0)	73.4 (0.1)
MMD	86.9 (0.9)	66.8 (0.8)
FEUDA	**89.3** (0.5)	**73.5** (0.8)

Table 5: Performance of FEUDA with causal language modeling and decoder-only architectures, with the GPT-2 medium model on the Apparel $\to$ Movies domain pair. FEUDA remains powerful, and improves performance on the source and target domains.

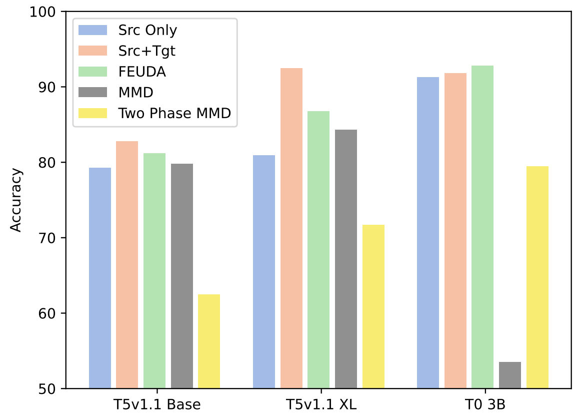

Figure 3: Target domain accuracy of FEUDA across different models, in a 256-shot learning setup on the Apparel $\to$ Movies domain pair. We see FEUDA retaining strong performance on the target domain across models.

Table 4 shows FEUDA beats MMD across different adaptation methods. We also note that fine-tuning yields slightly better performance on both domains for all UDA methods.

Few-Shot Learning For the following experiments, we assume access to k labeled source domain examples. For FEUDA, we assume access to the full unlabeled dataset in both domains for MLM pre-training, and k-shot access to labeled source data points for the second phase of supervised training. For a fair comparison, we also introduce a two-phase version of the MMD pipeline – the first phase minimizes MMD between unlabeled source and target domain embeddings (full data access), while the second phase optimizes supervised training on the source domain (k-shot).

Figure 3 (Table 13 in Appendix H) showcases FEUDA clearly outperforming both variants of MMD, across three different models. In Appendix H, we also repeat the analysis from Section 6.2 and find that, like with the full-data setting, semantic and background features are required for classification on the unlabeled target domain. However in this setting, downstream classification is aided more by the masking of informative words, rather than uninformative words.

7 Conclusion

We introduce the setting of prompt-based UDA, where the discriminative prediction is converted into a generative task. We then study the necessity of continued pre-training and domain-invariance based methods for UDA by introducing FEUDA, a "frustratingly easy" prompt-based UDA method. FEUDA involves training an auto-regressive LM on the unlabeled source and target data through the MLM task, followed by supervised training on the labeled source data. Across various datasets, models, adaptation methods, and few-shot settings, FEUDA is competitive with strong UDA methods that promote domain invariance. We also investigate the impact of continued pre-training on the UDA setup. We discover that the MLM task aids the model in learning both semantic and background knowledge of a domain, both of which are required for effective classification on the unlabeled target domain. We also discover that high masking rates are harmful to only the target domain, shedding new light on prior studies that study masking rates in single-domain setups. We hope our study will inspire future investigations in the prompt-based UDA setting.

Ethical Considerations

Our project aims to improve the reliability and safety of language models, which can be fragile under distribution shift (Ribeiro et al., 2020) and incur great costs over incorrect predictions (Ulmer et al., 2020; Zhang et al., 2021a). By improving performance over distributions without access to labelled data, our method can lead to direct benefits in a wide array of real world applications.

Our study does not involve any human subjects or violation of legal compliance. We do not anticipate any potentially harmful consequences to our work. As detailed in Appendix C, all of our experiments are conducted using publicly available datasets. Our code shall be released for reproducibility. Through our study and releasing our code, we hope to raise stronger research and societal awareness toward the problem of unsupervised domain adaptation in natural language processing.

Limitations and Risks

In our study, we consider a class of PEFT methods that involve inserting learnable parameters between the layers of the model. Other classes of PEFT methods were not considered. However, we use Adapters and He et al. (2022) have shown connections between the method with Prefix Tuning (Li and Liang, 2021) and LoRA (Hu et al., 2022).

Due to the high variance across runs in PEFT-based learning, we note that the performance can vary significantly across random seeds. We attempt to make our findings reproducible by averaging every experiment over 3 seeds. Taking environmental costs into consideration, we reduce our computational budget by running a majority of our experiments with a smaller-sized model. Learning with larger models is discussed in Appendix F.

References

Nachman Aronszajn. 1950. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404.

Stephen Bach, Victor Sanh, Zheng Xin Yong, Albert Webson, Colin Raffel, Nihal V Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Févry, et al. 2022. Promptsource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104.

Eyal Ben-David, Carmel Rabinovitz, and Roi Reichart. 2020. Perl: Pivot-based domain adaptation for pre-trained deep contextualized embedding models. *Transactions of the Association for Computational Linguistics*, 8:504–521.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning*, 79:151–175.

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

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

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. *Advances in neural information processing systems*, 29.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ruisi Cai, Zhenyu Zhang, and Zhangyang Wang. 2023. Robust weight signatures: Gaining robustness as easy as patching weights? *arXiv preprint arXiv:2302.12480*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.

Robert M Fano. 1961. Transmission of information: A statistical theory of communications. *American Journal of Physics*, 29(11):793–794.

Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. 2023. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12799–12807.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Pengfei Ge, Chuan-Xian Ren, Xiao-Lin Xu, and Hong Yan. 2023. Unsupervised domain adaptation via deep conditional adaptation network. *Pattern Recognition*, 134:109088.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773.

Xu Guo, Boyang Li, and Han Yu. 2022. Improving the sample efficiency of prompt tuning with domain adaptation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3523–3537, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Chia-Chien Hung, Lukas Lange, and Jannik Strötgen. 2023. Tada: Efficient task-agnostic domain adaptation for transformers. *arXiv preprint arXiv:2305.12717*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Constantinos Karouzos, Georgios Paraskevopoulos, and Alexandros Potamianos. 2021. Udalm: Unsupervised domain adaptation through language modeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2579–2590.

Abhinav Ramesh Kashyap, Devamanyu Hazarika, Min-Yen Kan, and Roger Zimmermann. 2021. Domain divergences: A survey and empirical analysis. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1830–1849.

Seungwon Kim, Alex Shum, Nathan Susanj, and Jonathan Hilgart. 2021. Revisiting pretraining with adapters. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 90–99.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. What's in a domain? learning domain-robust text representations using adversarial training. *arXiv preprint arXiv:1805.06088*.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR.

Quanyu Long, Tianze Luo, Wenya Wang, and Sinno Pan. 2022. Domain confused contrastive learning for unsupervised domain adaptation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2982–2995.

Bhavitvya Malik, Abhinav Ramesh Kashyap, Min-Yen Kan, and Soujanya Poria. 2023. Udapter-efficient domain adaptation using adapters. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2241–2255.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th international conference on World wide web*, pages 751–760.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in nlp—a survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, et al. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352.

Kendrick Shen, Robbie M Jones, Ananya Kumar, Sang Michael Xie, Jeff Z HaoChen, Tengyu Ma, and Percy Liang. 2022. Connect, not collapse: Explaining contrastive learning for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 19847–19878. PMLR.

Baochen Sun, Jiashi Feng, and Kate Saenko. 2017. Correlation alignment for unsupervised domain adaptation. *Domain adaptation in computer vision applications*, pages 153–171.

Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. 2019. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Nghia Ngo Trung, Linh Ngo Van, and Thien Huu Nguyen. 2022. Unsupervised domain adaptation for text classification via meta self-paced learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4741–4752.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep domain confusion: Maximizing for domain invariance. *conference on computer vision and pattern recognition*.

Dennis Ulmer, Lotta Meijerink, and Giovanni Cinà. 2020. Trust issues: Uncertainty estimation does not enable reliable ood detection on medical tabular data. In *Machine Learning for Health*, pages 341–354. PMLR.

Rheeya Uppaal, Junjie Hu, and Yixuan Li. 2023. Is fine-tuning needed? pre-trained language models are near perfect for out-of-domain detection. In *Annual Meeting of the Association for Computational Linguistics*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. 2023. Should you mask 15% in masked language modeling? In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2985–3000, Dubrovnik, Croatia. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Garrett Wilson and Diane J Cook. 2020. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(5):1–46.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pages 23965–23998. PMLR.

Hui Wu and Xiaodong Shi. 2022. Adversarial soft prompt tuning for cross-domain sentiment analysis. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2438–2447.

Oliver Zhang, Jean-Benoit Delbrouck, and Daniel L Rubin. 2021a. Out of distribution detection for medical images. In *Uncertainty for Safe Utilization of Machine Learning in Medical Imaging, and Perinatal Imaging, Placental and Preterm Image Analysis*, pages 102–111. Springer.

Rongsheng Zhang, Yinhe Zheng, Xiaoxi Mao, and Minlie Huang. 2021b. Unsupervised domain adaptation with adapter. *Proceedings of the Workshop on Efficient Natural Language and Speech Processing*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. 2019. On learning invariant representations for domain adaptation. In *International conference on machine learning*, pages 7523–7532. PMLR.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on knowledge and Data Engineering*, 17(11):1529–1541.

Yftah Ziser and Roi Reichart. 2018. Pivot based language modeling for improved neural domain adaptation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1241–1251.

A PEFT Frameworks

The framework proposed in Section 4 is generic, and can be applied to full-model fine-tuning. However, we additionally explore learning in a parameter-efficient setup. Specifically, we use two instantiations in our implementations: Adapters (Houlsby et al., 2019) and $(IA)^3$ (Liu et al., 2022).

$(IA)^3$ is a state of the art PEFT learning method, and uses around a tenth of learnable parameters compared to popular methods like Adapters. $(IA)^3$ works by element-wise multiplication (i.e. rescaling) of the model's activations against a learned vector. In this case, the set of learnable parameters ϕ is a set of vectors $\{l_{\mathrm{v}}, l_{\mathrm{k}}, l_{\mathrm{ff}}\}$ applied to each attention mechanism and feed-forward layer as,

$$h = \sigma\left(\frac{Q(l_{\mathrm{k}} \odot K^T)}{\sqrt{d_k}}\right)(l_{\mathrm{v}} \odot V)$$

$$h = (l_{\mathrm{ff}} \odot \gamma(W_1 x) W_2)$$

Here, K, Q and V are the key, query and value representations used in an attention block, and W_1 and W_2 are the weights in the feed-forward layer following an attention block. $l_{\mathrm{k}} \in \mathbb{R}^{d_k}$, $l_{\mathrm{v}} \in \mathbb{R}^{d_v}$, $l_{\mathrm{ff}} \in \mathbb{R}^{d_{\mathrm{ff}}}$, σ is the softmax function while γ is any non-linearity.

Intuitively, each vector l simply learns weights measuring the importance of each feature in an activation of the pre-trained model, for the specific downstream task the model is trained on.

Adapters are a popularly used and high performing PEFT framework, and He et al. (2022) have shown equivalence in the operations applied by Adapters, Prefix Tuning (Li and Liang, 2021) and LoRA (Hu et al., 2022).

Adapters work by adding small learnable modules between transformer layers. Specifically, down and up projections $W_{\mathrm{down}} \in \mathbb{R}^{d \times r}$ and $W_{\mathrm{up}} \in \mathbb{R}^{r \times d}$ are learnt such that $\phi = \{W_{\mathrm{up}}, W_{\mathrm{down}}\}$. A residual connection and non-linearity γ is added at every layer,

$$h = h + \gamma(hW_{\mathrm{down}})W_{\mathrm{up}}$$

B Single Phase MLM Training

Our proposed approach in Section 4 involves two stages of training, which is more expensive than standard single phase UDA approaches. In this section, we propose a single training phase variant to FEUDA, and show that it performs similarly to the original method. We use the two phase pipeline in our experiments in the main paper, but note that the single and two phase pipelines are interchangeable.

We simply replace the two phase training with a joint multi-task objective as follows,

$$\mathcal{L}(\mathcal{D}, \mathcal{D}_{\mathrm{src}}; \theta) = \frac{1}{|D|}\frac{1}{|D_{\mathrm{src}}|}\sum_{x' \in \mathcal{D}}\sum_{(x,y) \in \mathcal{D}_{\mathrm{src}}} (\lambda\, l(\mathbb{C}(x,y);\theta) + (1-\lambda)\, l(\mathbb{M}(x');\theta))$$

where l is the cross-entropy loss defined in Eq. (1), and $\mathbb{M}$ and $\mathbb{C}$ are the templates defined in Section 4. λ is the adaptation factor which gradually changes from 0 to 1 over the course of training. This results in the model being trained almost exclusively on the MLM task early on in training, and the CLS task towards the end of training.

Table 6 compares the performance of the single phase and two phase variants of FEUDA. We also compare with a vanilla joint single phase objective, where λ is fixed at 0.5 through training (called Single Phase Vanilla). The performance of the single and two phase variants are almost identical, and either can be used interchangeably. In comparison, the vanilla single phase method is significantly weaker on the target domain.

Method	Accuracy	
	Source	Target
Two Phase	93.7 (0.3)	**83.3** (0.9)
Singe Phase	93.5 (0.4)	**83.3** (0.5)
Singe Phase Vanilla	93.6 (0.1)	75.0 (5.7)

Table 6: Comparison of single and two-phase variants of FEUDA, on the Apparel → Movies domain pair. The single and two phase variants are almost identical in performance.

C Preparation of Evaluation Benchmarks

We use two classification datasets, with 5 domains each. This results in a total of 40 pairs of source and target domains. For brevity, we include results of 24 domain pairs in the main paper, and the remaining 16 in Appendix E. For both datasets, we use the train, validation and test splits from (Malik et al., 2023). More statistics about each dataset is available in Table 7. The listed datasets are intended for research purposes only. We do not make any commercial use of them.

MNLI The Multigenre Natural Language Inference (MNLI) corpus (Williams et al., 2018) contains sentence pairs across multiple genres: Travel (T), Fiction (F), Government (G), Slate (S) and Telephone (Te). The NLI task involves classifying every premise-hypothesis sentence pair as Entailment, Neutral or Contradiction.

Amazon The Multi Domain Sentiment Analysis Dataset (Blitzer et al., 2007) contains Amazon product reviews for different type of products. We use reviews from the Apparel (A), Baby (B), Books (Bo), Cameras (C) and Movies (M) domains. Each review is labelled as positive or negative.

Dataset	**Language**	**License**	**Statistics per Domain** Train	Val	Test
MNLI	English	cc-by-4.0	69600*	7735**	1945
Amazon	English	cc-by-4.0	1440	160	400

Table 7: Artifacts used in our study. The dataset statistics report the values used in our study.
* All domains contain approximately 69,600 examples. The exception is the Telephone domain, with 75,013 examples.
** All domains contain 7735 validation examples, except for Slate and Telephone, which contain 7731 and 8336 examples respectively.

D Details on Implementation

Models and Implementation We use T5v1.1, T0 and GPT-2 from the HuggingFace library[4], and use PyTorch Lightning[5] to train our models. We use the codebase of Liu et al. (2022)[6] for implementations of PEFT methods.

Training We use the default hyperparameters from Liu et al. (2022), except for batch size and training duration. We perform a grid search for these values. We train each training phase for 30,000 steps on MNLI and 15,000 steps on the Amazon dataset, with a batch size of 8. For the T5v1.1 XL and T0 models (3B parameters each), we use a batch size of 1. We train with Adam and use a learning rate of 0.003. We set the maximum sequence length to 256 tokens. We use length normalization during evaluation, as proposed by Liu et al. (2022). For each experiment, we report the mean and standard deviation across 3 runs.

[4]https://github.com/huggingface/transformers
[5]https://lightning.ai/docs/pytorch/latest/
[6]https://github.com/r-three/t-few

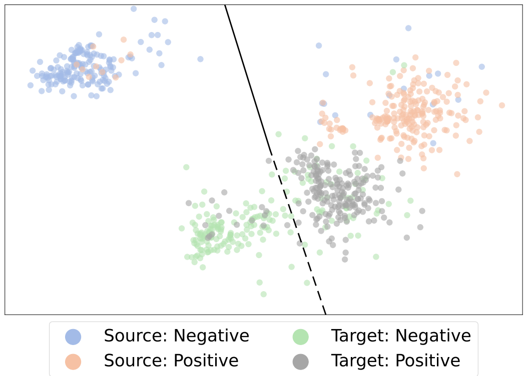

Figure 4: UMap visualizations of sentence embeddings from the Apparel → Movies data pair, using the T5v1.1 base model and (IA)^3 PEFT method. Despite not promoting domain-invariance, FEUDA learns sentence embeddings that are separable by class labels, regardless of the domain of these sentences. The classification hyperplane for the source domain has been imagined as a solid line for illustration purposes, and its extension to the target domain is shown as a dashed line.

Computations Using the (IA)^3 PEFT framework, training the T5v1.1 Base model (60 million parameters) for 15,000 steps takes approximately two hours on a single NVIDIA RTX A6000 GPU. The T5v1.1 XL model and T0 model (3 billion parameters) take approximately 8 hours for 15,000 steps of training. For reproducibility, each experiment is repeated thrice, with changing random seeds. In total, we run 540 experiments with the Base model and 72 experiments with the larger models. This results in a total compute time of approximately 2400 GPU hours.

E Results with Amazon Dataset

Table 8 shows the performance of our proposed approach on the Amazon Product Reviews dataset. On average, FEUDA is competitive with the state of the art MMD method from Malik et al. (2023).

We confirm this by checking for a significant difference in the performance of FEUDA and MMD on the 20 dataset pairs. The Mann-Whitney U test and Student's t-test both resulted in non-significant p-values of 0.5516 and 0.8316, confirming the hypothesis that there is no significant difference between FEUDA and MMD on the Amazon dataset. However, on the MNLI dataset, where all domains have larger gaps, both significant tests showed a significant difference between FEUDA and MMD, with FEUDA being more powerful.

Source → Target	Source Accuracy				Target Accuracy			
	Src Only	Src+Tgt	MMD	FEUDA	Src Only	Src+Tgt	MMD	FEUDA
A->B	93.7 (0.1)	93.8 (0.3)	94.3 (0.2)	93.1 (0.3)	93.3 (0.4)	94.7 (0.2)	93.8 (0.3)	**93.9** (0.3)
A->B	93.7 (0.1)	94.2 (0.1)	93.8 (0.1)	92.8 (0.6)	90.8 (0.6)	94.3 (0.4)	**92.5** (1.1)	90.2 (1.2)
A->C	93.7 (0.1)	93.4 (0.3)	95.0 (0.0)	93.9 (0.5)	91.9 (0.1)	95.0 (0.2)	91.8 (0.5)	**92.1** (0.5)
A->M	93.7 (0.1)	94.1 (0.3)	94.7 (0.3)	93.5 (0.4)	81.3 (1.4)	85.8 (0.5)	81.3 (0.6)	**83.3** (0.5)
B->A	95.5 (0.2)	94.8 (0.1)	95.8 (0.5)	95.2 (0.2)	93.0 (0.4)	93.4 (0.3)	93.3 (0.2)	**93.4** (0.4)
B->Bo	95.5 (0.2)	94.9 (0.1)	95.8 (0.2)	94.3 (0.3)	93.0 (0.9)	94.7 (0.7)	**93.8** (0.3)	92.2 (0.1)
B->C	95.5 (0.2)	95.2 (0.2)	96.0 (0.8)	94.6 (0.7)	93.1 (0.3)	94.7 (0.8)	**93.4** (0.1)	92.1 (0.3)
B->M	95.5 (0.2)	94.5 (0.4)	96.0 (0.3)	93.9 (0.7)	82.0 (0.5)	85.3 (0.2)	81.3 (0.7)	**82.8** (0.2)
Bo->A	94.3 (0.6)	94.7 (0.3)	93.8 (0.4)	91.9 (0.5)	92.3 (0.4)	94.6 (0.3)	**91.6** (0.5)	91.3 (0.2)
Bo->B	94.3 (0.6)	94.4 (0.5)	94.9 (0.7)	92.2 (0.3)	93.6 (0.3)	94.8 (0.2)	**92.9** (0.6)	90.9 (0.2)
Bo->C	94.3 (0.6)	93.8 (0.7)	94.7 (0.5)	91.6 (0.3)	89.3 (1.1)	94.3 (0.2)	89.8 (0.1)	**90.3** (0.4)
Bo->M	94.3 (0.6)	94.3 (0.3)	94.2 (0.4)	91.3 (0.4)	81.7 (0.8)	85.5 (0.9)	**84.6** (0.7)	80.1 (1.2)
C->A	93.9 (0.2)	94.6 (0.1)	93.5 (0.2)	93.4 (0.5)	92.1 (0.3)	93.4 (0.4)	92.3 (0.3)	**92.5** (0.6)
C->B	93.9 (0.2)	94.3 (0.9)	93.3 (0.7)	93.3 (0.6)	94.0 (0.2)	95.0 (0.6)	**94.1** (0.1)	92.1 (0.2)
C->Bo	93.9 (0.2)	93.9 (0.1)	62.3 (0.0)	92.3 (0.2)	91.1 (0.4)	93.9 (0.8)	**91.3** (0.5)	89.0 (0.1)
C->M	93.9 (0.2)	94.1 (0.3)	92.9 (0.9)	93.4 (0.5)	82.3 (1.1)	85.8 (0.1)	**81.5** (0.7)	79.7 (1.2)
M->A	85.5 (0.2)	85.6 (0.7)	86.3 (0.6)	83.3 (0.6)	89.8 (0.8)	94.2 (0.7)	89.1 (1.4)	**90.1** (0.5)
M->B	85.5 (0.2)	86.1 (0.4)	78.3 (11.1)	83.7 (0.3)	91.7 (0.5)	95.3 (0.5)	81.0 (16.1)	**89.9** (1.2)
M->Bo	85.5 (0.2)	84.6 (0.8)	76.4 (13.7)	83.8 (0.0)	92.7 (0.1)	94.1 (0.4)	80.5 (18.6)	**91.5** (0.0)
M->C	85.5 (0.2)	86.1 (0.7)	87.0 (0.0)	84.6 (0.6)	90.1 (0.3)	94.3 (0.5)	**90.5** (0.0)	89.7 (0.3)

Table 8: Comparison of FEUDA and MMD by classification accuracy on the Amazon Product Review dataset, using the T5v1.1 base model, and $(IA)^3$ PEFT method. FEUDA is competitive with MMD on average. For the target domain, the highest values between FEUDA and MMD have been marked in bold. Cases where Src Only outperforms both FEUDA and MMD have been underlined.

FEUDA can learn representations that generalize across domains To better understand the improved UDA performance, we visualize the sentence embeddings learned by FEUDA in Figure 4. Using UMap (McInnes et al., 2018), the figure visualizes embeddings for the Apparel→Movies domain pair from the Amazon Product Review dataset. We see that FEUDA learns sentence embeddings that generalize across domains. For illustration, we draw a black line that cuts across both source and target domains. Note that the solid line suggests that there exists a classification hyperplane learned on the source labeled data (in blue and green). The same classifier can be potentially used to separate target data (in gray and orange). The visualization suggests that FEUDA achieves competitive UDA results without having to explicitly promote domain-invariant representations.

F Learning with Larger Models

The use of high capacity language models, which are too compute intensive for traditional fine-tuning, originally encouraged the use of prompting, and prompting is now common with learning from large language models (Brown et al., 2020; Schick and Schütze, 2021a,b; Gao et al., 2021). Since our approach makes use of prompting, we investigate the performance of FEUDA with such models.

We experiment with two large models: T5v1.1 XL (3 billion parameters) and T0 (3 billion parameters). T0 is optimized for zero-shot generalization by training on supervised prompts. We use the Apparel (A) → Movies (M) domain pair from the Amazon Reviews dataset. Table 9 shows the performance gap between FEUDA and MMD increasing with larger models. In the case of T0, we see particularly poor performance with MMD. This may be due to the fact that the task of minimizing the divergence between embeddings is highly different from the tasks a model is trained on during instruction tuning.

G More On Analyzing the Impact of MLM on UDA

Table 10 accompanies the results from Figure 1, which shows the impact of masking sequences at inference. Words are selected for masking based on their their "informativeness", measured by their PMI to the inference class label. The performance of the model is best with the original unmasked

Model	Src Only		Src+Tgt		MMD		FEUDA	
	Source	Target	Source	Target	Source	Target	Source	Target
T5 v1.1 Base	93.7 (0.1)	81.3 (1.4)	94.1 (0.3)	85.8 (0.5)	93.4 (0.2)	78.6 (1.3)	93.5 (0.4)	**83.3** (0.5)
T5 v1.1 XL	95.4 (0.2)	89.1 (0.8)	95.3 (0.6)	93.0 (0.5)	74.8 (15.6)	65.2 (9.5)	95.1 (0.2)	**92.0** (1.5)
T0 3B	95.5 (0.4)	91.3 (0.2)	95.5 (0.2)	92.2 (0.7)	52.1 (1.1)	51.8 (0.8)	95.5 (0.4)	**93.8** (0.4)

Table 9: Performance of FEUDA across the T5v1.1 Base (60 million parameters), T5v1.1 XL (3 billion parameters) and T0 (3 billion parameters) models on the Apparel → Movies domain pair. We report the mean and standard deviation over 3 runs. The performance gap between FEUDA and MMD increases with larger models.

sequences, indicating the presence of *both* informative and uninformative words are essential for strong classification performance.

To isolate any effects of PEFT methods or pre-training data, we repeat the analysis from Table 3 with fine-tuning Flan-T5 and notice a similar trend in Table 11.

Table 12 accompanies results from Figure 2, which show the impact of varying masking rates on FEUDA. Using the T5v1.1 base model, we train FEUDA using varying random masking rates on the Apparel → Movies domain pair, and report the mean and standard deviation over three runs. With high masking rates, the performance on the source domain is largely maintained, but the performance on the target domain rapidly deteriorates.

Method	Accuracy	
	Source	Target
Original	**93.5**	**83.3**
Informative Masking	88.0	78.8
Uninformative Masking	92.0	79.0

Table 10: Impact of masking at inference. We evaluate FEUDA on the Apparel → Movies domain pair, and select words for masking based on their "informativeness" to the classification task. The performance of the model is best with the original unmasked sequences, indicating the presence of *both* informative and uninformative words are essential for strong classification performance.

Masking Strategy	Accuracy	
	Source	Target
Random	**95.8** (0.0)	**86.8** (0.3)
Informative	93.9 (0.6)	85.3 (0.3)
Uninformative	95.0 (0.0)	84.8 (0.1)

Table 11: Impact of word selection for masking during training. Using Flan-T5 base and no PEFT methods, we find that masking words at random is more powerful than selectively masking informative or uninformative words. This indicates that the model requires *both* semantic features (learnt through masking informative words) and background features (learnt through masking uninformative words) for classification on the unlabelled target domain.

Masking Rate	Accuracy	
	Source	Target
5%	92.8 (0.8)	78.8 (1.8)
15%	**93.5** (0.4)	**83.3** (0.5)
30%	92.8 (0.6)	78.8 (1.4)
60%	92.5 (0.9)	71.0 (3.0)
90%	92.3 (0.5)	70.4 (1.5)

Table 12: Impact of Masking Rate on FEUDA. We train FEUDA using varying random masking rates on the Apparel → Movies domain pair. With high masking rates, the performance on the source domain is largely maintained, but the performance on the target domain rapidly deteriorates.

H Learning in a Few-Shot Setup

Classification Accuracy Table 13 accompanies Figure 3 (Section 6.3), showing the 256-shot performance of FEUDA and other baselines, across model sizes.

Impact of Masked Words We extend the analysis on the impact of masked word selection from Section 6.2 to the few-shot setting in Table 14, where we compare the impact of masking informative or uninformative words. We also consider two different few-shot setups: one with access to the full unlabelled datasets in phase 1 pre-training, and another where even the unlabelled data is few-shot. Similar to the full-data setting, random masking remains most powerful, indicating that both semantic and background features are necessary for effective classification on the unlabelled domain. However, unlike the full-data setting where informative and uninformative masking are comparable, in the few-shot setting, informative masking is significantly more useful.

Model	Src Only		Src+Tgt		MMD		Two Phase MMD		FEUDA	
	Source	Target	Source	Target	Source	Target	Source	Target	Source	Target
T5v1.1 Base	92.0 (0.4)	79.3 (0.8)	93.2 (0.1)	82.8 (0.6)	80.1 (0.7)	62.5 (0.7)	92.3 (0.5)	79.8 (1.4)	90.1 (0.5)	81.2 (0.7)
T5v1.1 XL	91.1 (6.1)	80.9 (8.4)	95.7 (0.1)	92.5 (0.4)	87.3 (6.8)	71.7 (7.8)	92.2 (0.1)	84.3 (0.9)	95.2 (0.3)	86.8 (2.2)
T0 3B	95.3 (0.3)	91.3 (0.3)	95.6 (0.3)	91.8 (0.6)	91.3 (5.1)	79.5 (6.7)	54.0 (4.8)	53.5 (0.4)	95.8 (0.1)	92.8 (0.2)

Table 13: Performance of FEUDA across different models, in a 256-shot learning setup on the Apparel → Movies domain pair. We see FEUDA retaining strong performance on the target domain across models.

Phase 1 Data	Masking Strategy	Accuracy	
		Source	Target
256 Shot	Random	**91.0** (0.9)	**78.1** (2.4)
	Informative	90.4 (0.5)	76.0 (0.7)
	Uninformative	89.6 (1.2)	73.5 (1.6)
Full Data	Random	90.1 (0.5)	**81.2** (0.7)
	Informative	**91.8** (0.5)	78.0 (0.9)
	Uninformative	89.3 (0.5)	72.8 (1.1)

Table 14: Impact of word selection for masking, in a 256-shot learning setup. We evaluate FEUDA on the Apparel → Movies domain pair, and select words for masking based on their "informativeness" to the classification task. Random masking is most powerful for the target domain, indicating that both semantic and background features are necessary for effective classification on the unlabelled domain. However, informative masking is significantly more useful than uninformative masking.

I Comparison With More Baselines

Our main comparisons are made with the MMD based method proposed by (Malik et al., 2023), as it is a recent and powerful UDA method that outperforms other popular invariance based UDA approaches like DANN (Ganin et al., 2016; Li et al., 2018) and DSN (Bousmalis et al., 2016).

For a more comprehensive evaluation, we include a comparison with more baselines in this section. Using the T5v1.1 base model with $(IA)^3$ on the Amazon Apparel→Movies data pair, we include a comparison with DANN, which is the most widely used UDA method in NLP (Ramponi and Plank, 2020). However, DANN has been shown to be highly unstable in prior work, and we thus also minimize alternate measures of domain divergence based on Wasserstein distance and second order statistics (CORAL) (Sun et al., 2017). Our method is competitive with all baselines.

Additionally, with an emerging class of weight interpolation based methods, we make a comparison with task vector arithmetic (Ilharco et al., 2022). The use of task vectors with PEFT has been unexplored in the literature, and we find that the method does not work with IA3. With fully fine-tuned models, the method improves in performance, but is still weaker than FEUDA.

Method	Accuracy	
	Source	Target
FEUDA	93.7 (0.3)	**83.3** (0.9)
MMD	**94.7** (0.3)	81.3 (0.6)
DANN	53.5 (2.3)	52.3 (1.7)
CORAL	94.6 (0.2)	80.9 (0.4)
Wasserstein	94.5 (0.2)	82.5 (0.4)
Task Vectors	46.3 (0.3)	48.0 (0.7)
Task Vectors (fine-tuning)	93.0 (0.2)	69.0 (0.4)

Table 15: Comparison of FEUDA with more baselines, using the T5v1.1 base model and $(IA)^3$ PEFT method on the Apparel→Movies pair from the Amazon review dataset. For task vectors, we include versions with $(IA)^3$ as well as full fine-tuning. FEUDA outperforms all baselines.

J More About MMD

The Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) measures the difference between first order moments of variables in a Reproducing Kernel Hilbert Space (Aronszajn, 1950). Multiple lines of work have shown that minimizing divergence measures like MMD, when combined with auxiliary task-specific loss functions, results in training instabilities and vanishing gradients (Kashyap et al., 2021; Han and Eisenstein, 2019).

We also note that as minimizing MMD does not use any label information, there is a possibility for embeddings of the target domain to be aligned with the closest source domain class cluster. For example, Figure 5 shows us a setting where both classes of the target domain (shown in green and gray) are mapped to the cluster of negative class source embeddings (shown in blue).

We compare variants of the MMD method in Table 16 and show that the loss is sensitive to small changes in the loss design. Specifically we compare the MMD method used in the main paper with:

- MMD over Logits: Measures the MMD between the logits of source and target domains, instead of using intermediate model outputs.

- Fixed Weight MMD: Instead of the multi-task loss for the MMD reduction and classification tasks, we use fixed weights for both tasks[7].

- Two Phase MMD: The first training phase is used to minimize MMD between source and target embeddings, while the second phase is used to train the model for classification on the source domain.

FEUDA remains more powerful than all variants.

Method	**Accuracy**	
	Source	Target
FEUDA	93.7 (0.3)	**83.3** (0.9)
MMD	**94.7** (0.3)	81.3 (0.6)
MMD over Logits	95.0 (0.2)	81.3 (0.7)
Fixed Weight MMD	93.4 (0.2)	78.6 (1.3)
Two Phase MMD	90.1 (0.1)	68.7 (2.0)

Table 16: Comparison of variants of minimizing MMD, on the Apparel $\rightarrow$ Movies domain pair. FEUDA remains more powerful than all variants.

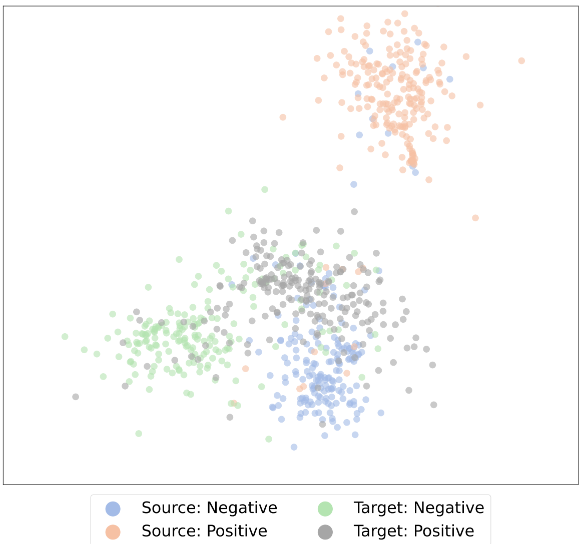

Figure 5: UMap visualizations of sentence embeddings from the Apparel $\rightarrow$ Movies data pair, using the T5v1.1 base model and (IA)^3 PEFT method. Training with MMD risks stability issues, and all embeddings from the target domain can be mapped to the closest source class cluster. This results in poor classification performance on the target domain.

[7]For the weighted loss, $\mathcal{L}_{\text{CLS}} + 3\ \mathcal{L}_{\text{MMD}}$ was found to be the best performing.