Latent-Optimized Adversarial Neural Transfer for Sarcasm Detection

Xu Guo, Boyang Li*, Han Yu and **Chunyan Miao***
School of Computer Science and Engineering,
Nanyang Technological University, Singapore
{xu008, han.yu, boyang.li, ascymiao}@ntu.edu.sg

Abstract

The existence of multiple datasets for sarcasm detection prompts us to apply transfer learning to exploit their commonality. The adversarial neural transfer (ANT) framework utilizes multiple loss terms that encourage the source-domain and the target-domain feature distributions to be similar while optimizing for domain-specific performance. However, these objectives may be in conflict, which can lead to optimization difficulties and sometimes diminished transfer. We propose a generalized latent optimization strategy that allows different losses to accommodate each other and improves training dynamics. The proposed method outperforms transfer learning and meta-learning baselines. In particular, we achieve 10.02% absolute performance gain over the previous state of the art on the iSarcasm dataset.

1 Introduction

Sarcastic language is commonly found in social media posts (González-Ibáñez et al., 2011; Maynard and Greenwood, 2014), forum discussions (Khodak et al., 2018a), product reviews (Davidov et al., 2010; Filatova, 2012) and everyday conversations (Gibbs, 2000). Detecting sarcasm is an integral part of creative language understanding (Veale et al., 2019) and online opinion mining (Kannangara, 2018). Due to highly contextualized expressions, detecting sarcasm is a challenging task, even for humans (Fox Tree et al., 2020).

A challenge specific to sarcasm detection is the difficulty in acquiring ground-truth annotations. Human-annotated datasets (Filatova, 2012; Riloff et al., 2013; Van Hee et al., 2018; Oprea and Magdy, 2020) usually contain only a few thousand texts, resulting in many small datasets. In comparison, automatic data collection using distant supervision signals like hashtags (Ptáček et al., 2014; Bamman and Smith, 2015; Joshi et al., 2015) yielded substantially larger datasets. Nevertheless, the automatic approach also led to label noise. For example, Oprea and Magdy (2020) found nearly half of the tweets with sarcasm hashtags in one dataset are not sarcastic.

The existence of diverse datasets and data collection methods prompts us to exploit their commonality through transfer learning. Specifically, we transfer knowledge learned from large and noisy datasets to improve sarcasm detection on small human-annotated datasets that serve as effective performance benchmarks.

Adversarial neural transfer (ANT) (Ganin and Lempitsky, 2015; Liu et al., 2017; Kim et al., 2017; Kamath et al., 2019) employs an adversarial setup where the network learns to make the shared feature distributions of the source domain and the target domain as similar as possible, while simultaneously optimizing for domain-specific performance. However, as the domain-specific losses promote the use of domain-specific features, these training objectives may compete with each other implicitly. This leads to optimization difficulties and potentially degenerate cases where the domain-specific classifiers ignore the shared features and no meaningful transfer occurs between domains.

To cope with this issue, we propose Latent-Optimized Adversarial Neural Transfer (LOANT). The latent optimization strategy can be understood with analogies to to one-step look-ahead during gradient descent and Model-Agnostic Meta Learning (Finn et al., 2017). By forcing domain-specific losses to accommodate the negative domain discrimination loss, it improves training dynamics (Balduzzi et al., 2018).

With LOANT, we achieve 10.02% absolute improvement over the previous state of the art on the iSarcasm dataset (Oprea and Magdy, 2020) and 3.08% improvement on SemEval-18 dataset (Van Hee et al., 2018). Over four sets of transfer learning experiments, latent optimization on aver-

* Corresponding authors

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5394–5407
June 6–11, 2021.

age brings 3.42% improvement in F-score over traditional adversarial neural transfer and 4.83% over a similar training strategy from Model-Agnostic Meta Learning (MAML) (Finn et al., 2017). In contrast, traditional ANT brings an average of only 0.9% F-score improvement over non-adversarial multi-task learning. The results demonstrates that LOANT can effectively perform knowledge transfer for the task of sarcasm detection and suggests that the proposed latent optimization strategy enables the collaboration among the ANT losses during optimization.

Our contributions can be summarized as follows:

1. Inspired by the existence of multiple small sarcasm datasets, we propose to use transfer learning to bridge dataset differences. To the best of our knowledge, this is the first study of transfer learning between different sarcasm detection datasets.

2. We propose LOANT, a novel latent-optimized adversarial neural transfer model for cross-domain sarcasm detection. By conducting stochastic gradient descent (SGD) with one-step look-ahead, LOANT outperforms traditional adversarial neural transfer, multi-task learning, and meta-learning baselines, and establishes a new state-of-the-art F-score of 46.41%. The code and datasets are available at https://github.com/guoxuxu/LOANT.

2 Related Work

2.1 Sarcasm Detection

Acquiring large and reliable datasets has been a persistent challenge for computational detection of sarcasm. Due to the cost of annotation, manually labeled datasets (Walker et al., 2012; Riloff et al., 2013; Wallace et al., 2014; Abercrombie and Hovy, 2016; Oraby et al., 2016; Van Hee et al., 2018; Oprea and Magdy, 2020) typically contain only a few thousand texts. Automatic crawling (Ptáček et al., 2014; Bamman and Smith, 2015; Joshi et al., 2015; Khodak et al., 2018b) using hashtags or markers yields substantially more texts, but the results are understandably more noisy. As a case study, after examining the dataset of Riloff et al. (2013), Oprea and Magdy (2020) found that nearly half of tweets with sarcasm hashtags are not sarcastic. In this paper, we evaluate performance on the manually labeled datasets, which are relatively clean and can serve as good benchmarks, and transfer the knowledge learned from automatically collected datasets.

Traditional sarcasm detection includes methods based on rules (Tepperman et al., 2006) and lexical (Kreuz and Caucci, 2007) and pragmatic patterns (González-Ibánez et al., 2011). Context-aware methods (Rajadesingan et al., 2015; Bamman and Smith, 2015) make use of contexts, such as the author, the audience, and the environment, to enrich feature representations.

Deep learning techniques for sarcasm detection employ convolutional networks (Ghosh and Veale, 2016), recurrent neural networks (Zhang et al., 2016; Felbo et al., 2017; Wu et al., 2018), attention (Tay et al., 2018), and pooling (Xiong et al., 2019) operations. Amir et al. (2016) incorporate historic information for each Twitter user. Cai et al. (2019) consider the images that accompany tweets and Mishra et al. (2017) utilize readers' gaze patterns. To the best of our knowledge, no prior work has explored transfer learning between different sarcasm datasets.

2.2 Adversarial Transfer Learning

As a transfer learning technique, multi-task learning (MTL) allows related tasks or similar domains to inform each other and has been a powerful technique for NLP (Collobert et al., 2011; Yang et al., 2017; Aharoni et al., 2019; Guo et al., 2019; Raffel et al., 2020). However, MTL does not always lead to performance improvements (Alonso and Plank, 2017; Bingel and Søgaard, 2017; Changpinyo et al., 2018; Clark et al., 2019).

Theoretical analysis (Ben-David et al., 2010) indicates that a key factor for the success of transfer is to reduce the divergence between the feature spaces of the domains. Ganin and Lempitsky (2015) propose to minimize domain differences via a GAN-like setup, where a domain discriminator network learns to distinguish between features from two domains and a feature extraction network learns to produce indistinguishable features, which are conducive to transfer learning.

Similar adversarial setups (Liu et al., 2017; Kim et al., 2017) have been adopted for many NLP tasks, such as sentiment analysis (Chen et al., 2018; Liu et al., 2018), satirical news detection (McHardy et al., 2019), detection of duplicate questions (Kamath et al., 2019), named entity recognition (Zhou

et al., 2019), and QA (Yu et al., 2018).

However, as shown in our experiments, adding the domain discriminator to MTL does not always result in improved performance. We attribute this to the implicit competition between the negative domain discrimination loss and the domain-specific losses, which causes difficulties in optimization. In this paper, we improve the training dynamics of adversarial transfer learning using latent optimization on BERT features.

2.3 Meta-Learning and Latent Optimization

The idea of coordinating gradient updates of different and competing losses using gradient descent with look-ahead has been explored in Latent-optimized Generative Adversarial Network (LOGAN) (Wu et al., 2019b,a), Symplectic Gradient Adjustment (Balduzzi et al., 2018; Gemp and Mahadevan, 2019), Unrolled GAN (Metz et al., 2016), Model-Agnostic Meta Learning (Finn et al., 2017) and extragradient (Azizian et al., 2020). The difference between LOGAN and other techniques is that the LOGAN computes the derivative of the randomly sampled latent input, whereas other methods compute the second-order derivative in the model parameter space.

In this paper, we generalize latent optimization from GANs to multi-task learning, where the adversarial loss is complemented by domain-specific task losses. In addition, we apply latent optimization on the output of the BERT module, which differs from the optimization of the random latent variable in LOGAN. As large pretrained masked language models (PMLMs) gain prominence in NLP, latent optimization avoids gradient computation on the parameters of enormous PMLMs, providing reduction in running time and memory usage.

3 The LOANT Method

In supervised transfer learning, we assume labeled data for both the source domain and the target domain are available. The source domain dataset D_s comprises of data points in the format of (x_s, y_s) and the target domain dataset D_t comprises of data points in the format of (x_t, y_t). The labels y_s and y_t are one-hot vectors. The task of supervised cross-domain sarcasm detection can be formulated as learning a target-domain function $f_t(x_t)$ that predict correct labels for unseen x_t.

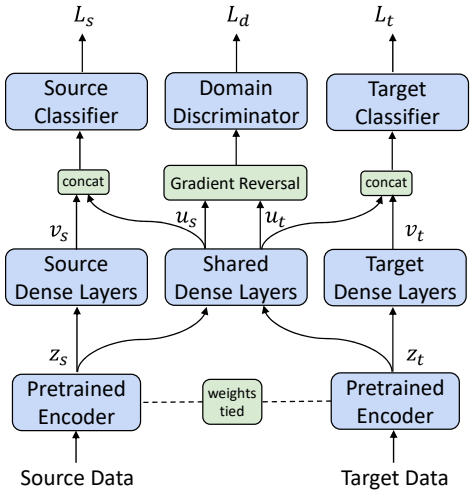


Figure 1: Network architecture of the Adversarial Neural Transfer model.

3.1 Model Architecture

Fig. 1 shows the model architecture for adversarial neural transfer (ANT) (Liu et al., 2017; Kamath et al., 2019; Kim et al., 2017). We use a large pretrained neural network, BERT (Devlin et al., 2019), as the sentence encoder, though the architecture is not tied to BERT and can use other pretrained encoders. We denote the parameters of the BERT encoder as w_b, and its output for data in the source domain and the target domain as $z_s \in \mathbb{R}^D$ and $z_t \in \mathbb{R}^D$ respectively. We denote this encoder operation as

$$z_s = E(x_s, w_b),\ z_t = E(x_t, w_b) \tag{1}$$

On top of these outputs, we apply domain-specific dense layers to create domain-specific features v_s, v_t and shared dense layers to create shared features u_s, u_t. We use w_s, w_t, and w_{sh} to denote the parameters for the source dense layers, the target dense layers, and the shared dense layers.

The concatenation of features $[v_s, u_s]$ is fed to the source-domain classifier, parameterized by θ_s; $[v_t, u_t]$ is fed to the target-domain classifier, parameterized by θ_t. The two classifiers categorize the tweets into sarcastic and non-sarcastic and are trained using cross-entropy. For reasons that will become apparent later, we make explicit the reliance on z_s and z_t:

$$\begin{aligned} \mathcal{L}_s(z_s) &= -\sum_i y_{s,i} \log p(\hat{y}_{s,i}|z_s), \\ \mathcal{L}_t(z_t) &= -\sum_i y_{t,i} \log p(\hat{y}_{t,i}|z_t), \end{aligned} \tag{2}$$

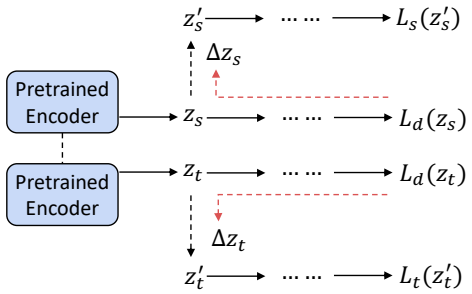


Figure 2: Schematic of the latent optimization strategy. The solid black arrows indicate the forward pass and the dotted red arrows indicate the backward pass.

where $\hat{y}_s$ and $\hat{y}_t$ are the predicted labels and i is the index of the vector components.

Simultaneously, the domain discriminator learns to distinguish the features u_s and u_t as coming from different domains. The domain discriminator is parameterized by θ_d. It is trained to minimize the domain classification loss,

$$\mathcal{L}_d(z_t, z_s) = -\log p(0|u_s) - \log p(1|u_t). \quad (3)$$

Through the use of the gradient reversal layer, the shared dense layers and the feature encoder maximizes the domain classification loss, so that the shared features u_s and u_t become indistinguishable and conducive to transfer learning. In summary, the network weights $w_b, w_s, w_t, w_{sh}, \theta_s, \theta_t$ are trained to minimize the following joint loss,

$$\mathcal{L}^{\text{ANT}} = \mathcal{L}_s(z_s) + \mathcal{L}_t(z_t) - \mathcal{L}_d(z_t, z_s), \quad (4)$$

whereas θ_d is trained to minimize $\mathcal{L}_d(z_t, z_s)$.

It is worth noting that the effects of three loss terms in Eq. 4 on the shared parameters w_{sh} and w_b may be competing with each other. This is because optimizing sarcasm detection in one domain will encourage the network to extract domain-specific features, whereas the domain discrimination loss constrains the network to avoid such features. It is possible for the competition to result in degenerate scenarios. For example, the shared features u_s and u_t may become indistinguishable but also do not correlate with the labels y_s and y_t. The domain classifiers may ignore the shared features u_s and u_t and hence no transfer happens. To cope with this issue, we introduce a latent optimization strategy that forces domain-specific losses to accommodate the domain discrimination loss.

3.2 Latent Representation Optimization

We now introduce the latent representation optimization strategy. First, we perform one step of stochastic gradient descent on $-\mathcal{L}_d$ on the encoded features z_s and z_t with learning rate γ,

$$z'_s = z_s + \gamma \frac{\partial \mathcal{L}_d(z_s, z_t)}{\partial z_s}, \quad (5)$$

$$z'_t = z_t + \gamma \frac{\partial \mathcal{L}_d(z_s, z_t)}{\partial z_t}. \quad (6)$$

We emphasize that this is a *descent* step because we are minimizing $-\mathcal{L}_d$.

After that, we use the updated z'_s and z'_t in the computation of the losses

$$\mathcal{L}_s^{\text{LO}}(z_s, z'_s) = \mathcal{L}_s(z_s) + \mathcal{L}_s(z'_s), \quad (7)$$

$$\mathcal{L}_t^{\text{LO}}(z_t, z'_t) = \mathcal{L}_t(z_t) + \mathcal{L}_t(z'_t). \quad (8)$$

The new joint objective hence becomes

$$\begin{aligned}\mathcal{L}^{\text{LO}} = \mathcal{L}_s^{\text{LO}}(z_s, z'_s) + \mathcal{L}_t^{\text{LO}}(z_t, z'_t) \\ - \mathcal{L}_d(z_s, z_t),\end{aligned} \quad (9)$$

which is optimized using regular stochastic gradient descent (SGD) on $w_b, w_s, w_t, w_{sh}, \theta_s$, and θ_t.

Here we show the general case of gradient computation. Consider any weight vector w in the neural network. Equations 5 and 6 introduce two intermediate variables z'_s and z'_t, which are a function of the model parameter w. Therefore, we perform SGD using the following total derivative

$$\begin{aligned}\frac{\mathrm{d}\mathcal{L}^{\text{LO}}}{\mathrm{d}w} = \frac{\partial \mathcal{L}^{\text{LO}}}{\partial w} + \frac{\partial \mathcal{L}_s^{\text{LO}}(z'_s)}{\partial z'_s}\frac{\partial z'_s}{\partial w} \\ + \frac{\partial \mathcal{L}_t^{\text{LO}}(z'_t)}{\partial z'_t}\frac{\partial z'_t}{\partial w}.\end{aligned} \quad (10)$$

where

$$\begin{aligned}\frac{\partial z'_s}{\partial w} = \frac{\partial z_s}{\partial w} + \gamma \frac{\partial^2 \mathcal{L}_d(z_s)}{\partial z_s\, \partial w} \\ \frac{\partial z'_t}{\partial w} = \frac{\partial z_t}{\partial w} + \gamma \frac{\partial^2 \mathcal{L}_d(z_t)}{\partial z_t\, \partial w}\end{aligned} \quad (11)$$

For every network parameter other than the encoder weight w_b, $\partial z / \partial w$ is zero. The second-order derivative $\partial^2 \mathcal{L}_d(z) / \partial z\, \partial w$ is difficult to compute due to the high dimensionality of w. Since γ is usually very small, we adopt a first-order approximation and directly set the second-order derivative to zero. Letting $\phi_s = [w_s, \theta_s]$ and $\phi_t = [w_t, \theta_t]$, we now show the total derivatives for all network

Algorithm 1: Training of LOANT

Input: source data (x_s, y_s), target data (x_t, y_t), learning rate γ

Initialize model parameters w

repeat

  Sample N batches of data pairs

  for *i = 1* **to** N **do**

    Compute forward loss $\mathcal{L}_s$, $\mathcal{L}_t$, $\mathcal{L}_d$;

    Compute $\triangle z_s = \frac{\partial \mathcal{L}_d(z_s)}{\partial z_s}$ and $\triangle z_t = \frac{\partial \mathcal{L}_d(z_t)}{\partial z_t}$;

    Update the latent representations $z'_s = z_s + \gamma \triangle z_s$ and $z'_t = z_t + \gamma \triangle z_t$;

    Compute the new joint loss $\mathcal{L}^{\text{LO}} = \mathcal{L}_s^{\text{LO}} + \mathcal{L}_t^{\text{LO}} - \mathcal{L}_d$;

    Update w using gradient descent.

until *the maximum training epoch*

parameters:

$$\frac{\mathrm{d}\mathcal{L}^{\text{LO}}}{\mathrm{d}w_b} = \frac{\partial \mathcal{L}^{\text{ANT}}}{\partial w_b} + \frac{\partial \mathcal{L}_s(z'_s)}{\partial w_b} + \frac{\partial \mathcal{L}_t(z'_t)}{\partial w_b} + \frac{\partial \mathcal{L}_s(z'_s)}{\partial z'_s}\frac{\partial z_s}{\partial w_b} + \frac{\partial \mathcal{L}_t(z'_t)}{\partial z'_t}\frac{\partial z_t}{\partial w_b} \tag{12}$$

$$\frac{\mathrm{d}\mathcal{L}^{\text{LO}}}{\mathrm{d}w_{sh}} = \frac{\partial \mathcal{L}^{\text{ANT}}}{\partial w_{sh}} + \frac{\partial \mathcal{L}_s(z'_s)}{\partial w_{sh}} + \frac{\partial \mathcal{L}_t(z'_t)}{\partial w_{sh}} \tag{13}$$

$$\frac{\mathrm{d}\mathcal{L}^{\text{LO}}}{\mathrm{d}\phi_s} = \frac{\partial \mathcal{L}_s(z_s)}{\partial \phi_s} + \frac{\partial \mathcal{L}_s(z'_s)}{\partial \phi_s}$$

$$\frac{\mathrm{d}\mathcal{L}^{\text{LO}}}{\mathrm{d}\phi_t} = \frac{\partial \mathcal{L}_t(z_t)}{\partial \phi_t} + \frac{\partial \mathcal{L}_t(z'_t)}{\partial \phi_t} \tag{14}$$

$$\frac{\mathrm{d}\mathcal{L}^{\text{LO}}}{\mathrm{d}\theta_d} = \frac{\partial \mathcal{L}_d(z_s, z_t)}{\partial \theta_d}$$

More details can be found in Appendix A. Fig. 2 illustrates the latent optimization process. Algorithm 1 shows the LOANT algorithm.

3.3 Understanding LOANT

To better understand the LOANT algorithm, we relate LOANT to the extragradient technique and Model-Agnostic Meta Learning (Finn et al., 2017).

The vanilla gradient descent (GD) algorithm follows the direction along which the function value decreases the fastest. However, when facing an ill-conditioned problem like the one in Fig. 3, GD is known to exhibit slow convergence because the local gradients are close to being orthogonal to the direction of the local optimum.

For comparison with LOANT, we consider the extragradient (EG) method (Korpelevich, 1976; Azizian et al., 2020) that uses the following update rule when optimizing the function $f(w)$ with respect to w,

$$w \leftarrow w - \eta \frac{\mathrm{d} f(w - \gamma \frac{\partial f(w)}{\partial w})}{\mathrm{d}w}. \tag{15}$$

Similar to LOANT, we can adopt a first-order approximation to EG if we set the Hessian term to zero in the total derivative. Instead of optimizing the immediate function value $f(w)$, this method optimizes $f(w - \gamma \frac{\partial f}{\partial w})$, which is the function value after one more GD step. This can be understood as looking one step ahead along the optimization trajectory. In the contour diagrams of Fig. 3, we show the optimization of a 2-dimensional quadratic function. This simple example showcases how the ability to look one step ahead can improve optimization in pathological loss landscapes. We motivate the nested optimization of LOANT by drawing an analogy between EG and LOANT.

It is worth noting that LOANT differs from the EG update rule in important ways. Specifically, in EG the inner GD step and the outer GD step are performed on the same function $f(\cdot)$, whereas LOANT performs the inner step on $\mathcal{L}_d$ and the outer step on $\mathcal{L}_s$ or $\mathcal{L}_t$.

For a similar idea with multiple losses, we turn to MAML (Finn et al., 2017). In MAML, there are K tasks with losses $\mathcal{L}_1, \ldots, \mathcal{L}_k, \ldots, \mathcal{L}_K$. On every task, we perform a one-step SGD update to the model parameter $w \in \mathbb{R}^L$,

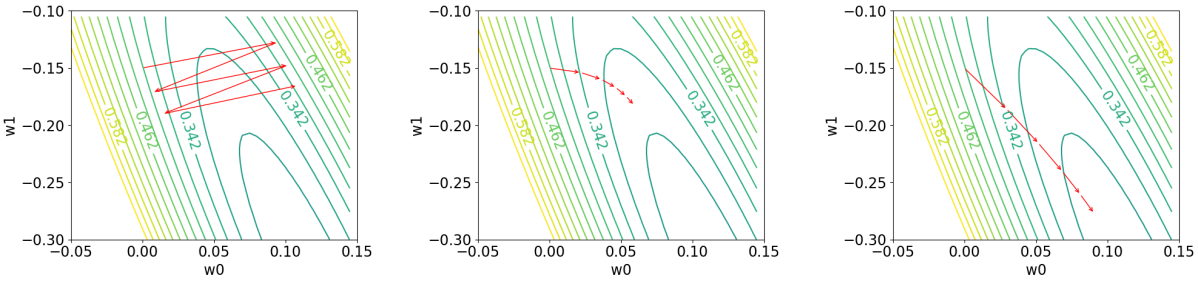
$$w_{T_k} = w - \gamma \frac{\partial \mathcal{L}_k(w)}{\partial w}. \tag{16}$$

After going through K tasks, the actual update to w is calculated using the parameters w_{T_k},

$$w \leftarrow w - \eta \frac{1}{K} \sum_k \frac{\mathrm{d}\mathcal{L}_k(w_{T_k})}{\mathrm{d}w}. \tag{17}$$

Utilizing the idea of look ahead, in MAML we update w so that subsequent optimization on any single task or combination of tasks would achieve good results.

Adversarial neural transfer has three tasks, the source-domain and target-domain classifications and the negative discriminator loss. The updates performed by LOANT in Eq. 5 and 6 are similar to MAML's look-ahead update in Eq. 16. Specifically, when we update model parameters using the gradient from the total loss $\mathcal{L}^{\text{LO}}$, we prepare for the next descent step on $-\mathcal{L}_d$. Therefore, LOANT can



(a) Vanilla gradient descent, which exhibits a zigzag trajectory. $\eta = 0.025$.

(b) First-order extragradient, which sets the Hessian term to zero. $\eta = 0.025$. $\gamma = 0.01$.

(c) Full-Hessian extragradient, which finds a direct path to the local minimum, enabling a large learning rate $\eta = 0.1$.

Figure 3: Minimization of a 2D function $f(w) = w^\top Aw + b^\top w + c$. A is positive definite and has a condition number of 40. The initial point is $(0, -0.15)$. The red arrows show the trajectory of w. The look-ahead capability of extragradient finds a much more direct path to the local minimum than vanilla gradient descent.

be understood as forcing domain-specific losses to accommodate the domain discrimination loss and mitigating their competition.

LOANT differs from MAML since, in the inner update, LOANT updates the sentence-level features z_s and z_t instead of the model parameters w. As z_s and z_t are usually of much smaller dimensions than w, this leads to accelerated training and reduced memory footprint. For example, in the BERT-base model (Devlin et al., 2019), L is 110 million and D is 768. Within the regular range of batch size B, $BD \ll L$. In the experiments, we verify the benefits of LOANT in terms of accuracy and time and space complexity.

4 Experiments

4.1 Datasets

We conduct four cross-domain sarcasm detection experiments by transferring from an automatically collected dataset to a manually annotated dataset. The two automatically collected datasets include Ptáček (Ptáček et al., 2014) and Ghosh[1] (Ghosh and Veale, 2016), which treat tweets having particular hastags such as `#sarcastic`, `#sarcasm` or `#not` as sarcastic and others as not sarcastic. We crawled the Ptáček dataset using the NLTK API[2] according to the tweet ids published online[3].

The two manually annotated datasets include SemEval-18[4] (Van Hee et al., 2018) and iSarcasm (Oprea and Magdy, 2020). SemEval-18 consists of both sarcastic and ironic tweets supervised by third-party annotators and thus is used for *perceived* sarcasm detection. The iSarcasm dataset contains tweets written by participants of an online survey and thus is an example of *intended* sarcasm detection.

Dataset	Train	Val	Test	% Sarcasm
Ptáček	51009	5668	6298	49.50%
Ghosh	33373	3709	4121	44.84%
SemEval-18	3398	378	780	49.12%
iSarcasm	3116	347	887	17.62%

Table 1: Dataset statistics, including number of samples in each split and the proportion of sarcastic texts.

Table 1 summarizes the statistics of the four datasets. The SemEval-18 dataset is balanced while the iSarcasm dataset is imbalanced. The two source datasets are more than ten times the size of the target datasets. For all datasets, we use the predefined test set and use a random 10% split of the training set as the development set.

We preprocessed all datasets using the lexical normalization tool for tweets from Baziotis et al. (2017). We cleaned the four datasets by dropping all the duplicate tweets within and across datasets, and trimmed the texts to a maximum length of 100. To deal with class imbalance, we performed upsampling on the target-domain datasets, so that both the sarcastic and non-sarcastic classes have the same size as source domain datasets.

4.2 Baselines

We compare LOANT with several competitive single-task and multi-task baselines.

MIARN (Tay et al., 2018): A state-of-the-art short

[1] https://github.com/AniSkywalker/SarcasmDetection/tree/master/resource

[2] http://www.nltk.org/howto/twitter.html

[3] http://liks.fav.zcu.cz/sarcasm/

[4] https://github.com/Cyvhee/SemEval2018-Task3/tree/master/datasets

text sarcasm detection model ranked top-1 on the iSarcam dataset. The model is a co-attention based LSTM model which uses the word embeddings pretrained on Twitter data[5].

Dense-LSTM (Wu et al., 2018): A state-of-the-art single-task sarcasm detection model ranked top-1 on the SemEval-18 dataset. The model is a densely connected LSTM network consisting of four Bi-LSTM layers and the word embeddings pretrained on two Twitter datasets.

BERT: We finetune the BERT model (Devlin et al., 2019) with an additional simple classifier directly on the target dataset.

S-BERT is a two-stage finetuning of the BERT model. We first finetune BERT on the source dataset and the best model is selected for further fine-tuning on the target dataset.

MTL: We implemented a multi-task learning (MTL) model, which has the same architecture as LOANT except that the domain discriminator is removed. We use BERT as the shared text encoding network.

MTL+LO: In this baseline, we applied latent optimization to MTL. As MTL does not have the adversarial discriminator, we use the domain-specific losses to optimize latent representations:

$$z'_s = z_s - \gamma \frac{\partial \mathcal{L}_s(z_s)}{\partial z_s} \tag{18}$$

$$z'_t = z_t - \gamma \frac{\partial \mathcal{L}_t(z_t)}{\partial z_t} \tag{19}$$

We use the above to replace Equations 5 and 6 and keep the rest training steps unchanged. This model is compared against MTL to study the effects of LO in non-adversarial training for cross-domain sarcasm detection.

ANT: This is the conventional adversarial neural transfer model with the same architecture as LOANT. The only difference is that we do not apply latent optimization. For fair comparisons, we use BERT as the text encoder.

ANT+MAML: In Section 3.3, we discussed the similarity between LO and MAML. Therefore, we create a baseline that uses a MAML-like strategy for encouraging the collaboration of different loss terms. Instead of optimizing the latent representation z_s and z_t, we first take a SGD step in the parameter space of w_b,

$$w'_b = w_b + \gamma \frac{\partial \mathcal{L}_d(z_s, z_t)}{\partial w_b}. \tag{20}$$

After that, we use w'_b to compute the gradients used in the actual updates to all model parameters, including w_b.

4.3 Experimental Settings

Model Settings. For all models using the BERT text encoder, we use the uncased version of the BERT-base model and take the 768-dimensional output from the last layer corresponding to the `[CLS]` token to represent a sentence. The BERT parameters are always shared between domains. For other network components, we randomly initialize the dense layers and classifiers. To minimize the effect of different random initializations, we generate the same set of initial parameters for each network component and use them across all baselines wherever possible.

The source dense layer, the shared dense layer, and the target dense layer are single linear layers with input size of 768 and output size of 768 followed by the tanh activation. The classifier in all models consists of two linear layers. The first linear layer has input size of 768×2 (taking both shared and domain-specific features) and output size of 768 followed by the ReLU activation. The second linear layer has input size 768 and output size 2 for binary classification. After that we apply the softmax operation. More details can be found in Appendix B.

Training Setting. We optimize all models using Adam (Kingma and Ba, 2014) with batch size of 128. We tune the learning rate (LR) on the development set from 1e-5 to 1e-4 in increments of 2e-5. To objectively assess the effects of latent optimization (LO), we first find the best LR for the base models such as ANT and MTL. After that, with the best LR unchanged, we apply LO to ANT and MTL. We use the cosine learning rate schedule for all models. All models are trained for 5 epochs on Nvidia V100 GPUs with 32GB of memory in mixed precision. Due to the large model size and pretrained weights of BERT, 5 epochs are sufficient for convergence.

Evaluation Metrics. Following (Wu et al., 2018; Van Hee et al., 2018; Oprea and Magdy, 2020), we select and compare models using the F-score on the sarcastic class in each dataset. We additionally

[5] https://nlp.stanford.edu/projects/glove/

	Target: SemEval-18				Target: iSarcasm			
	Model	F-score	Recall	Precision	Model	F-score	Recall	Precision
Single-task	Random†	0.3730	0.3730	0.3730	SIARN‡	0.3420	0.7820	0.2190
	Unigram SVM†	0.5890	0.6590	0.5320	MIARN‡	<u>0.3640</u>	0.7930	0.2360
	LSTM†	0.5260	0.4440	0.6450	LSTM‡	0.3360	0.7470	0.2170
	DenseLSTM *	<u>0.6510</u>	0.7106	0.6005	DenseLSTM‡	0.3180	0.2760	0.3750
	BERT	0.6626	0.7055	0.6246	BERT	0.3492	0.4904	0.2711
Source: Ptáče	S-BERT	0.6676	0.7055	0.6337	S-BERT	0.3710	0.5541	0.2788
	MTL	0.6404	0.7896	0.5386	MTL	0.3767	0.3503	0.4074
	ANT	0.6348	0.8187	0.5184	ANT	0.3857	0.5159	0.3079
	MTL+LO	0.6598	0.7346	0.5989	MTL+LO	0.4379	0.4267	0.4496
	ANT+MAML	0.6454	0.7540	0.5641	ANT+MAML	0.3951	0.5605	0.2923
	LOANT (ours)	**0.6702**	0.8025	0.5754	LOANT (ours)	**0.4642**	0.4968	0.4357
Source: Ghosh	S-BERT	0.6512	0.7766	0.5607	S-BERT	0.3383	0.5732	0.2400
	MTL	0.6525	0.7475	0.5789	MTL	0.3838	0.5159	0.3056
	ANT	0.6626	0.8899	0.5278	ANT	0.4063	0.4904	0.3468
	MTL+LO	0.6622	0.8058	0.5620	MTL+LO	0.3987	0.4012	0.3962
	ANT+MAML	0.6338	0.7281	0.5610	ANT+MAML	0.3589	0.4904	0.2830
	LOANT (ours)	**0.6818**	0.7734	0.6096	LOANT (ours)	**0.4101**	0.4649	0.3668

† Results reported in (Van Hee et al., 2018), * in (Wu et al., 2018) and ‡ in (Oprea and Magdy, 2020).

Table 2: Performance on the sarcastic class reported by single-task and multi-task models on the same test sets. The best performed F-score on the four groups of transfer learning are in bold. The best single task learning results are underlined.

report the corresponding Recall and Precision. In all our experiments, we use the development set for model selection and report their performance on the test set. To evaluate the efficiency of LOANT versus MAML-based training, we also compare their required GPU memory and average training time in each epoch. We compare models on the target domain datasets. Additional multi-domain performance can be found in Appendix C.

4.4 Comparison with the States of the Art

We compare LOANT with state-of-the-art methods on the SemEval-18 dataset (Van Hee et al., 2018) and the iSarcasm datast (Oprea and Magdy, 2020). Table 2 presents the test performance of LOANT and all baseline models. Our LOANT model consistently outperforms all single-task baselines by large margins. In particular, LOANT outperforms MIARN by 10.02% on iSarcasm (Oprea and Magdy, 2020) whereas the fine-tuned BERT achieved 1.48% lower than MIARN. On SemEval-18, the fine-tuned BERT achieves better test performance than other four single-task baselines. The results indicate that fine-tuning BERT, a popular baseline, does not always outperform the traditional LSTM networks specifically designed for the task. We hypothesize that the large BERT model can easily overfit the small datasets used, which highlights the challenge of sarcasm detection.

		SemEval-18	iSarcasm
	Model	RAM/Time	RAM/Time
Source: Ptáče	LOANT	1.01x/2.41x	1.01x/2.55x
	MTL+LO	1.01x/1.92x	1.01x/1.91x
	ANT	1.00x/1.00x	1.00x/1.00x
	ANT + MAML	1.99x/8.31x	1.93x/10.2x
Source: Ghosh	LOANT	1.01x/2.44x	1.01x/1.94x
	MTL+LO	1.01x/1.94x	1.01x/1.89x
	ANT	1.00x/1.00x	1.00x/1.00x
	ANT + MAML	1.99x/8.41x	1.93x/10.7x

Table 3: Running time and maximum memory footprint for different transfer learning methods.

4.5 Transfer Learning Performance

The middle and bottom sections of Table 2 present the test performance of six transfer learning models (S-BERT, MTL, ANT, MTL+LO, ANT+MAML, and LOANT) under four groups of transfer learning experiments. These models generally outperform the single-task models, demonstrating the importance of transfer learning. Among these, we have the following observations.

Effects of the Domain Discriminator. The performance differences between MTL and ANT can be explained by the addition of the domain discriminator, which encourages the shared features under the source domain and the target domain to have the same distributions. In the four pairs of experiments, ANT marginally outperforms MTL by an average of 0.9% F-score. In the Ptáček → SemEval-18 experiment, the domain discriminator causes F-score to decrease by 0.56%. Overall, the

benefits of the adversarial discriminator to transfer learning appear to be limited. As discussed earlier, the competition between the domain-specific losses and the negative domain discrimination loss may have contributed to the ineffectiveness of ANT.

Effects of Latent Optimization. We can observe the effects of LO by comparing ANT with LOANT and comparing MTL with MTL+LO. Note that in these experiments we adopted the best learning rates for the baseline models ANT and MTL rather than the latent-optimized models. On average, LOANT outperforms ANT by 3.42% in F-score and MTL+LO outperforms MTL by 2.63%, which clearly demonstrates the benefits provided by latent optimization.

Latent Space vs. Model Parameter Space. In the ANT+MAML baseline, we adopt a MAML-like optimization strategy, which performs the look-ahead in the BERT parameter space instead of the latent representation space. Interestingly, this strategy does not provide much improvements and on average performs 1.40% worse than ANT. LOANT clearly outperforms ANT+MAML.

In addition, optimization in the latent space also provides savings in computational time and space requirements. Table 3 shows the time and memory consumption for different transfer learning methods. Adding LO to ANT has minimal effects on the memory usage, but adding MAML nearly doubles the memory consumption. On average, ANT+MAML increases the running time of LOANT by 3.1 fold.

The Influence of Domain Divergence. In transfer learning, the test performance depends on the similarity between the domains. We thus investigate the dissimilarity between datasets using the Kullback–Leibler (KL) divergence between the unigram probability distributions,

$$d_{KL} = \sum_{g \in V} P_t(g) \log \frac{P_t(g)}{P_s(g)}. \qquad (21)$$

where $P_s(g)$ and $P_t(g)$ are the probabilities of unigram g for the source domain and target domain respectively. V is the vocabulary. Table 4 shows the results. Ptáček is more similar to the two target datasets than Ghosh. Among the two target datasets, iSarcasm is more similar to Ptáček than SemEval-18.

Comparing LOANT and ANT, we observe that the largest improvement, 7.85%, happens in the Ptáček → iSarcasm transfer where domain divergence is the smallest. The Ptáček → SemEval-18 transfer comes in second with 3.54%. Transferring from Ghosh yields smaller improvements. Further, we observe the same trend in the comparison between MTL+LO and MTL. The largest improvement brought by LO is 6.12% in the Ptáček → iSarcasm transfer. As one may expect, applying LO leads to greater performance gains when the two domains are more similar.

	SemEval-18	iSarcasm
Ptáček	0.1631	0.0521
Ghosh	0.2300	0.2217

Table 4: The KL divergence of word probability over the overlapped vocabulary for each pair of domains.

5 Conclusion

Transfer learning holds the promise for the effective utilization of multiple datasets for sarcasm detection. In this paper, we propose a latent optimization (LO) strategy for adversarial transfer learning for sarcasm detection. By providing look-ahead in the gradient updates, the LO technique allows multiple losses to accommodate each other. This proves to be particularly effective in adversarial transfer learning where the domain-specific losses and the adversarial loss potentially conflict with one another. With the proposed LOANT method, we set a new state of the art for the iSarcasm dataset. We hope the joint utilization of multiple datasets will contribute to the creation of contextualized semantic understanding that is necessary for successful sarcasm detection.

Acknowledgments

This research is supported by the National Research Foundation, Singapore under its the AI Singapore Programme (AISG2-RP-2020-019), NRF Investigatorship (NRF-NRFI05-2019-0002), and NRF Fellowship (NRF-NRFF13-2021-0006); the Joint NTU-WeBank Research Centre on Fintech (NWJ-2020-008); the Nanyang Assistant/Associate Professorships (NAP); the RIE 2020 Advanced Manufacturing and Engineering Programmatic Fund (A20G8b0102), Singapore; NTU-SDU-CFAIR (NSC-2019-011). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the funding agencies.

References

Gavin Abercrombie and Dirk Hovy. 2016. Putting sarcasm detection into context: The effects of class imbalance and manual labelling on supervised machine classification of twitter conversations. In *Proceedings of the ACL 2016 student research workshop*, pages 107–113.

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Héctor Martínez Alonso and Barbara Plank. 2017. When is multitask learning effective? semantic sequence prediction under varying data conditions. In *EACL*.

Silvio Amir, Byron C. Wallace, Hao Lyu, and Paula Carvalho Mário J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. In *CoNLL*.

Waïss Azizian, Ioannis Mitliagkas, Simon Lacoste-Julien, and Gauthier Gidel. 2020. A tight and unified analysis of gradient-based methods for a whole spectrum of games. In *AISTATS*.

David Balduzzi, Sébastien Racanière, James Martens, Jakob N. Foerster, Karl Tuyls, and Thore Graepel. 2018. The mechanics of n-player differentiable games. In *ICML*.

David Bamman and Noah A Smith. 2015. Contextualized sarcasm detection on twitter. In *Ninth international AAAI conference on web and social media*. Citeseer.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

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(1):151–175.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *EACL*.

Yitao Cai, Huiyu Cai, and Xiaojun Wan. 2019. Multi-modal sarcasm detection in Twitter with hierarchical fusion model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2506–2515, Florence, Italy. Association for Computational Linguistics.

Soravit Changpinyo, Hexiang Hu, and Fei Sha. 2018. Multi-task learning for sequence tagging: An empirical study. In *COLING*.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018. Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, 6:557–570.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D. Manning, and Quoc V. Le. 2019. Bam! born-again multi-task networks for natural language understanding. In *ACL*.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, page 107–116, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625, Copenhagen, Denmark. Association for Computational Linguistics.

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, Istanbul, Turkey.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Jean E. Fox Tree, J. Trevor D'Arcey, Alicia A. Hammond, and Alina S. Larson. 2020. The sarchasm: Sarcasm production and identification in spontaneous conversation. *Discourse Processes*, 57(5-6):507–533.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *ICML*, volume 37, pages 1180–1189.

Ian Gemp and Sridhar Mahadevan. 2019. Global convergence to the equilibrium of gans using variational inequalities. *arXiv 1808.01531*.

Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*, pages 161–169.

Raymond W. Gibbs. 2000. Irony in talk among friends. *Metaphor and Symbol*, 15(1-2):5–27.

Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in Twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 581–586.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2019. Autosem: Automatic task selection and mixing in multi-task learning. In *NAACL*.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762.

Anush Kamath, Sparsh Gupta, and Vitor Carvalho. 2019. Reversing gradients in adversarial domain adaptation for question deduplication and textual entailment tasks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5545–5550.

Sandeepa Kannangara. 2018. Mining twitter for fine-grained political opinion polarity classification, ideology detection and sarcasm detection. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 751–752.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018a. A large self-annotated corpus for sarcasm. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018b. A large self-annotated corpus for sarcasm. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Young-Bum Kim, Karl Stratos, and Dongchan Kim. 2017. Adversarial adaptation of synthetic or stale data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1297–1307.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

G. M. Korpelevich. 1976. An extragradient method for finding saddle points and for other problems. *Ekonomika i Matematicheskie Metody*, 12:747–756.

Roger Kreuz and Gina Caucci. 2007. Lexical influences on the perception of sarcasm. In *Proceedings of the Workshop on Computational Approaches to Figurative Language*, pages 1–4, Rochester, New York. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–10, Vancouver, Canada. Association for Computational Linguistics.

Qi Liu, Yue Zhang, and Jiangming Liu. 2018. Learning domain representation for multi-domain sentiment classification. 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 541–550.

Diana G Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *LREC 2014 Proceedings*. ELRA.

Robert McHardy, Heike Adel, and Roman Klinger. 2019. Adversarial training for satire detection: Controlling for confounding variables. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 660–665, Minneapolis, Minnesota. Association for Computational Linguistics.

Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. 2016. Unrolled generative adversarial networks. In *NeurIPS*.

Abhijit Mishra, Kuntal Dey, and Pushpak Bhattacharyya. 2017. Learning cognitive features from gaze data for sentiment and sarcasm classification using convolutional neural network. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Silviu Oprea and Walid Magdy. 2020. isarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn Walker. 2016. Creating and characterizing a diverse corpus of sarcasm in dialogue. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 31–41, Los Angeles. Association for Computational Linguistics.

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on Czech and English twitter. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 213–223.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the eighth ACM international conference on web search and data mining*, pages 97–106.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, LalindraDe Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentimentand negative situation. In *EMNLP*.

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1010–1020, Melbourne, Australia. Association for Computational Linguistics.

Joseph Tepperman, David Traum, and Shrikanth Narayanan. 2006. " yeah right": Sarcasm recognition for spoken dialogue systems. In *Ninth international conference on spoken language processing*.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Tony Veale, F Amílcar Cardoso, and Rafael Pérez y Pérez. 2019. Systematizing creativity: A computational view. In *Computational Creativity*, pages 1–19. Springer.

Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. 2012. A corpus for research on deliberation and debate. In *LREC*, volume 12, pages 812–817. Istanbul.

Byron C Wallace, Laura Kertz, Eugene Charniak, et al. 2014. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 512–516.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Junxin Liu, Zhigang Yuan, and Yongfeng Huang. 2018. Thu_ngn at SemEval-2018 task 3: Tweet irony detection with densely connected LSTM and multi-task learning. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 51–56.

Yan Wu, Jeff Donahue, David Balduzzi, Karen Simonyan, and Timothy Lillicrap. 2019a. Logan: Latent optimisation for generative adversarial networks. *arXiv preprint arXiv:1912.00953*.

Yan Wu, Mihaela Rosca, and Timothy Lillicrap. 2019b. Deep compressed sensing. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6850–6860. PMLR.

Tao Xiong, Peiran Zhang, Hongbo Zhu, and Yihui Yang. 2019. Sarcasm detection with self-matching networks and low-rank bilinear pooling. In *The World Wide Web Conference*, pages 2115–2124.

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

Jianfei Yu, Minghui Qiu, Jing Jiang, Jun Huang, Shuangyong Song, Wei Chu, and Haiqing Chen. 2018. Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, The 26th International Conference on Computational Linguistics: Technical Papers*, pages 2449–2460.

Joey Tianyi Zhou, Hao Zhang, Di Jin, Hongyuan Zhu, Meng Fang, Rick Siow Mong Goh, and Kenneth Kwok. 2019. Dual adversarial neural transfer for low-resource named entity recognition. In *ACL*, pages 3461–3471.

Appendix for "Latent-Optimized Adversarial Neural Transfer for Sarcasm Detection"

A First-order Approximation

Here we explain the gradients for the model parameters $w_b, w_{sh}, \phi_s, \phi_t$ and θ_d. Generically, we apply the first-order approximation by substituting Eq.11 into Eq. 10 and setting the Hessian to zero, which gives

$$\frac{\mathrm{d}\mathcal{L}^{\mathrm{LO}}}{\mathrm{d}w} = \frac{\partial\mathcal{L}^{\mathrm{LO}}}{\partial w} + \frac{\partial\mathcal{L}_s^{\mathrm{LO}}(z_s')}{\partial z_s'}\frac{\partial z_s}{\partial w} + \frac{\partial\mathcal{L}_t^{\mathrm{LO}}(z_t')}{\partial z_t'}\frac{\partial z_t}{\partial w}. \tag{22}$$

Note that z_s and z_t depend on only the parameter w_b. For the rest of the parameters, w_{sh}, ϕ_s, ϕ_t and θ_d, the partial derivatives $\frac{\partial z_s}{\partial w}$ and $\frac{\partial z_t}{\partial w}$ are zero.

Now we consider the joint objective (Eq. 9), which contains domain-specific classification losses produced by both the old latent vector z and the new latent vector z'. Thus, we derive at the generic formula

$$\begin{aligned}\frac{\partial\mathcal{L}^{\mathrm{LO}}}{\partial w} &= \frac{\partial\mathcal{L}_s^{\mathrm{LO}}}{\partial w} + \frac{\partial\mathcal{L}_t^{\mathrm{LO}}}{\partial w} - \frac{\partial\mathcal{L}_d}{\partial w} \\ &= \frac{\partial\mathcal{L}_s(z_s)}{\partial w} + \frac{\partial\mathcal{L}_t(z_t)}{\partial w} - \frac{\partial\mathcal{L}_d(z_s,z_t)}{\partial w} \\ &\quad + \frac{\partial\mathcal{L}_s(z_s')}{\partial w} + \frac{\partial\mathcal{L}_t(z_t')}{\partial w}\end{aligned} \tag{23}$$

By the same reasoning above, the total derivative of $\mathcal{L}^{\mathrm{LO}}$ against w_b is

$$\frac{\mathrm{d}\mathcal{L}^{\mathrm{LO}}}{\mathrm{d}w_b} = \frac{\partial\mathcal{L}^{\mathrm{LO}}}{\partial w_b} + \frac{\partial\mathcal{L}_s^{\mathrm{LO}}(z_s')}{\partial z_s'}\frac{\partial z_s}{\partial w_b} + \frac{\partial\mathcal{L}_t^{\mathrm{LO}}(z_t')}{\partial z_t'}\frac{\partial z_t}{\partial w_b} \tag{24}$$

$$\frac{\partial\mathcal{L}^{\mathrm{LO}}}{\partial w_b} = \frac{\partial\mathcal{L}_s(z_s)}{\partial w_b} + \frac{\partial\mathcal{L}_t(z_t)}{\partial w_b} - \frac{\partial\mathcal{L}_d(z_s,z_t)}{\partial w_b} + \frac{\partial\mathcal{L}_s(z_s')}{\partial w_b} + \frac{\partial\mathcal{L}_t(z_t')}{\partial w_b} \tag{25}$$

For the rest of the parameters, the computation is slightly different as they do not contribute to z_s and z_t.

$$\frac{\partial\mathcal{L}^{\mathrm{LO}}}{\partial w_{sh}} = \frac{\partial\mathcal{L}_s(z_s)}{\partial w_{sh}} + \frac{\partial\mathcal{L}_t(z_t)}{\partial w_{sh}} - \frac{\partial\mathcal{L}_d(z_s,z_t)}{\partial w_{sh}} + \frac{\partial\mathcal{L}_s(z_s')}{\partial w_{sh}} + \frac{\partial\mathcal{L}_t(z_t')}{\partial w_{sh}} \tag{26}$$

$$\frac{\partial\mathcal{L}^{\mathrm{LO}}}{\partial \phi_s} = \frac{\partial\mathcal{L}_s(z_s)}{\partial \phi_s} + \frac{\partial\mathcal{L}_s(z_s')}{\partial \phi_s} \tag{27}$$

$$\frac{\partial\mathcal{L}^{\mathrm{LO}}}{\partial \phi_t} = \frac{\partial\mathcal{L}_t(z_t)}{\partial \phi_t} + \frac{\partial\mathcal{L}_t(z_t')}{\partial \phi_t} \tag{28}$$

The parameter of the domain discriminator θ_d is updated to minimize $\mathcal{L}_d(z_s, z_t)$. This is in contrast to the rest of the model, which minimizes $-\mathcal{L}_d(z_s, z_t)$. The update rule for θ_d is

$$\theta_d \leftarrow \theta_d - \eta\frac{\partial\mathcal{L}_d(z_s,z_t)}{\partial\theta_d} \tag{29}$$

B Hyperparameters and Model Initialization

We set the batch size to 128 for all models and search for the optimal learning rate (LR) from 2e-5 to 1e-4 in increments of 2e-5 using the F-score on the development set. We show the best learning rates found in Table 5.

The best learning rate for fine-tuning BERT on SemEval-18 and iSarcasm is 4e-5. S-BERT model is finetuned twice, first on the source domain and then on the target domain. Thus, we search for one best learning rate for each finetuning using the source and target development sets respectively. The best first-round LR is 6e-05 for Ptáče and 8e-5 for Ghosh.

Other models, MTL, ANT and the LO-adpated versions are selected using the target development set. For a rigorous comparison, we use the best LR for ANT when training LOANT and the best LR for MTL when training MTL+LO.

We follow the released code[6] to implement the Gradient Reversal Layer. It is controlled by a schedule which gradually increases the weight of the gradients from the domain discrimination loss.

C Source Domain Performance

The original goal of the paper is to use automatically collected sarcasm datasets, which are large but *noisy*, to improve performance on human-annotated datasets, which are *clean* and provide good performance measure. That is why we provided only the target domain performance.

Upon close inspection, LOANT also improves the performance on the source domain, even though

[6] https://github.com/fungtion/DANN

Models	Ptáče → SemEval	Ghosh → SemEval	Ptáče → iSarcasm	Ghosh → iSarcasm
S-BERT	1e-4	1e-4	4e-5	2e-5
MTL	6e-4	8e-5	4e-5	1e-4
MTL+LO	6e-4	8e-5	4e-5	1e-4
ANT	2e-5	4e-5	2e-5	2e-5
ANT+MAML	2e-5	4e-5	2e-5	2e-5
LOANT	2e-5	4e-5	2e-5	2e-5

Table 5: Learning rate chosen by each model on the given search grid.

Domain	ANT	LOANT	MTL	MTL+LO
Ptacek	0.8307	**0.8484**	**0.8640**	0.8629
iSarcasm	0.3857	**0.4642**	0.3767	**0.4379**
Average	0.6082	**0.6563**	0.62035	**0.6504**
Ghosh	**0.7345**	0.6596	0.6609	**0.6688**
iSarcasm	0.4063	**0.4101**	0.3838	**0.3953**
Average	**0.5704**	0.5349	0.5224	**0.5321**
Ptacek	**0.8626**	0.8612	**0.8722**	0.8666
SemEval18	0.6348	**0.6702**	0.6404	**0.6598**
Average	0.7487	**0.7657**	0.7563	**0.7632**
Ghosh	0.7161	**0.7752**	**0.7700**	0.7579
SemEval18	0.6626	**0.6818**	0.6525	**0.6622**
Average	0.6894	**0.7285**	**0.7113**	0.7101

Table 6: Test F1 score for both domains using model selection on the target domain only.

Domain	ANT	LOANT	MTL	MTL+LO
Ptacek	0.8307	**0.8484**	**0.8640**	0.8629
iSarcasm	0.3857	**0.4642**	0.3767	**0.4379**
Average	0.6082	**0.6563**	0.6204	**0.6504**
Ghosh	0.7787	**0.7826**	**0.7859**	0.7807
iSarcasm	**0.3965**	0.3215	0.3764	**0.3953**
Average	**0.5876**	0.5521	**0.5812**	0.5880
Ptacek	0.8567	**0.8612**	**0.8720**	0.8632
SemEval18	0.6463	**0.6702**	0.6594	**0.6666**
Average	0.7515	**0.7657**	**0.7657**	0.7649
Ghosh	0.7919	**0.7962**	0.7672	**0.7884**
SemEval18	0.6427	**0.6490**	0.6357	**0.6442**
Average	0.7173	**0.7226**	0.7015	**0.7163**

Table 7: Test F1 score for both domains using model selection on the average F1 of the two domains.

model selection was performed on the target domain. Table 6shows the results.

In Table 7, we also show the results after model selection on both domains. Naturally, this might lead to slightly lowered target-domain performance than achieved by model selection on target domain only. Comparing LOANT with ANT, and MTL+LO with MTL, our results show that, in most cases, LO-based models improve both source and target domain F1. In particular, target domain F1 obtains more improvement than source domain F1. This suggests that LO provides benefits to knowledge transfer.