Transductive Adversarial Networks (TAN)

Sean Rowan [1]

Abstract

Transductive Adversarial Networks (TAN) is a novel domain-adaptation machine learning framework that is designed for learning a conditional probability distribution on unlabelled input data in a target domain, while also only having access to: (1) easily obtained labelled data from a related source domain, which may have a different conditional probability distribution than the target domain, and (2) a marginalised prior distribution on the labels for the target domain. TAN leverages a fully adversarial training procedure and a unique generator/encoder architecture which approximates the transductive combination of the available source- and target-domain data. A benefit of TAN is that it allows the distance between the source- and target-domain label-vector marginal probability distributions to be greater than 0 (i.e. different tasks across the source and target domains) whereas other domain-adaptation algorithms require this distance to equal 0 (i.e. a single task across the source and target domains). TAN can, however, still handle the latter case and is a more generalised approach to this case. Another benefit of TAN is that due to being a fully adversarial algorithm, it has the potential to accurately approximate highly complex distributions. Theoretical analysis demonstrates the viability of the TAN framework.

1. Introduction

The scenario of having access to a small amount of labeled data but a large amount of unlabelled data is a common one in practice. In an idealised learning situation, the conditional probability distribution between the input vector and the label vector across the sets of labeled and unlabelled data are equal. However, typically this does not occur in practice. Instead, the small amount of labeled data that is accessible is usually either significantly simpler than the encountered unlabelled data, or comes from a different domain with a different conditional probability distribution between its input vector and label vector. These two practical cases can be considered the same from a learning point-of-view as the latter practical case [8].

In the standard domain-adaptation learning scenario, it is expected that the labelled and unlabelled input vectors can be drawn from unique marginal probability distributions. However, it is required that the label-vector marginal probability distribution for the labelled and unlabelled sets of data are equal and match the available labelled set of data [8]. This is demonstrated in the following example involving the MNIST (hand-drawn digits) and SVHN (house numbers from Google StreetView images) datasets.

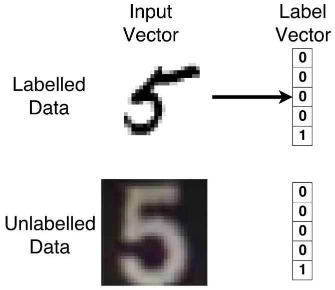

Figure 1. The standard domain-adaptation learning scenario where the label-vector marginal probability distributions across domains are expected to be equal. In this example learning scenario, the labelled data are pairs of both an image of a hand-drawn number from 1 through 5 and a 5-dimensional one-hot encoded vector that encodes the numerical representation of the input image. The unlabelled data are images of house numbers from 1 through 5. There are common features in the input vectors across domains that allow a domain-adaptation learning algorithm to assign labels to the unlabelled input vectors using the available labelled and unlabelled data.

Now consider a generalised domain-adaptation learning scenario where both the input-vector and the label-vector marginal probability distributions across domains are not expected to be equal. This generalised scenario motivates the design of TAN. The scenario is demonstrated in the following example involving the MNIST and SVHN datasets.

[1] University College London. Correspondence to: <sean.rowan.16@ucl.ac.uk>.

Code is available at:
https://github.com/sean-rowan/tan

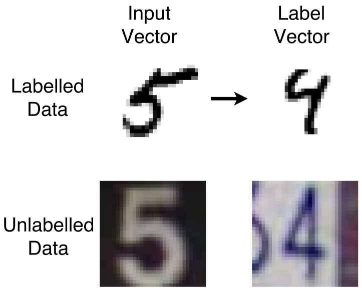

Figure 2. A generalised domain-adaptation learning scenario where both the input-vector and the label-vector marginal probability distributions across domains are not expected to be equal. In this example learning scenario, the labelled data are pairs of a hand-drawn single-digit image of an odd number and a hand-drawn single-digit image of the previous even number. The unlabelled data pairs are the same except they are images of house numbers.

This generalised domain-adaptation learning scenario is distinct from the style-transfer learning problem. In style transfer, learning occurs only on a single marginalised input vector across domains, and does not involve a corresponding conditional label vector, thus significantly reducing the scope of applications [7].

We now further motivate the usefulness of an algorithm that can learn a conditional probability distribution within the generalised domain-adaptation learning scenario with a real-world application. Consider the problem of human drug discovery. In a human drug discovery scenario, there is no data available about how an experimental drug molecule might bind to known human protein structures due to the difficulty in testing new drugs on live human subjects. However, there is ample data available on how an experimental drug molecule can bind to known yeast cell protein structures due to the free ability to test new drugs on these cells. In this scenario, the yeast cell experiments represent the source domain and the human experiments represent the target domain. The yeast cell protein structure is the input vector of the source domain and the experimental drug for yeast cells is the label vector of the source domain. Likewise, the human protein structure is the input vector of the target domain and the experimental drug for humans is the unknown label vector of the target domain. The learning goal is to generate a shortlist of potential candidate drugs for further human testing. Such an algorithm would be highly valuable in discovering new drugs that are suitable for humans with fewer human drug trials.

2. Related Work

A good overview of transfer learning research and terminology can be found at: [8], we follow this terminology.

There are two prior works that form the basis for the TAN framework: 1) Generative Adversarial Networks (GAN) [6] and 2) Adversarially Learned Inference (ALI) [4] (which is equivalent to BiGAN [3]). TAN leverages the general theoretical results from the GAN framework (the ALI framework leverages the GAN results as well) but utilises the ALI framework's training procedure as a component of the unique TAN algorithm.

In the GAN framework, a two-player zero-sum game between adversarial learning agents is played where one agent (the generator) learns to generate convincing fake data, while the other agent (the discriminator) learns to discern generated data from sampled data which comes from an unknown distribution. The generator learns a transfer function that converts an inputted Gaussian-noise vector into convincing data that matches the unknown data distribution on convergence of the adversarial game.

The ALI framework (and also the BiGAN framework) extends the GAN framework by simultaneously learning a reverse transfer function that maps inputted data back to the Gaussian-noise vector which generated it, allowing the ability to finely control the features of the generated data with interpolations in the Gaussian-noise vector. The ALI framework by itself does not allow the ability to learn conditional probability distributions on inputted conditional data pairs [2].

The GAN framework can directly learn conditional probability distributions, and has also previously been formulated for the standard domain-adaptation learning scenario [9]. However, the GAN framework and its variants are not suited to the generalised domain-adaptation learning scenario because the discriminator requires label-vectors[1] that come from the same marginalised probability distribution across the real and fake (i.e. source and target in this case) domains. TAN allows the label-vectors to come from different marginalised probability distributions across the real and fake domains.

Δ-GAN [5] is structurally similar to TAN in that it also leverages the GAN theoretical results and the ALI training procedure, however it is built for the more restrictive inductive transfer learning task and cannot handle the transductive transfer learning task. This means that Δ-GAN requires paired input/label training data in both the source and target domain, whereas TAN only requires paired input/label training data in the source domain, unlabelled input data in the target domain and a marginalised prior distribution on the label-vector distribution in the target domain.

[1] 'Label-vector' here means the actual data that the discriminator discerns as being real or fake, and not the real/fake label for the data inputted to the discriminator. Also, the input-vector for the domain-adaptation problem is inputted to the generator along with the Gaussian noise vector.

3. TAN Framework

In this section, we first outline the TAN probabilistic model, then we define the training procedure of this probabilistic model and finally we conclude with two proofs that establish the global optimality and convergence properties of the TAN framework.

TAN leverages both a GAN network and an ALI network, but with a shared generator across the two networks. The GAN network is trained normally and exclusively on the source-domain data. The ALI network is also trained normally but exclusively on the target-domain unlabelled input data and a prior on the target-domain label data. A unique training procedure which combines the two networks via the generator forces the shared generator and the ALI network's encoder to accommodate the statistics of both the source and target domain, and leads to convergence at a unique global optimum under mild assumptions (the exact same convexity assumptions from the original GAN framework formulation [6]). The trained encoder can then be used as an inference model on the target-domain unlabelled input-data.

3.1. Model

We first define our terms as follows. $\boldsymbol{x}$ is the input-vector and $\boldsymbol{z}$ is the label-vector. $p_s(\boldsymbol{x}, \boldsymbol{z})$ is the joint data distribution from the source domain. $p_s(\boldsymbol{z})$ is the marginalised label-vector data distribution from the source domain. $p_t(\boldsymbol{x})$ is the unlabelled input-vector data distribution from the target domain. $\widetilde{p}_t(\boldsymbol{z})$ is the label-vector prior distribution from the target domain. $G_x(\boldsymbol{x}|\boldsymbol{z};\theta_{gx})$ is the shared generator function. $G_z(\boldsymbol{z}|\boldsymbol{x};\theta_{gz})$ is the encoder function. y is the binary classification label of the inputted data to a discriminator. $y = 1$ for samples from the distribution that the discriminator learns to support. $D_s(y|\boldsymbol{x},\boldsymbol{z};\theta_{ds})$ is the source-domain discriminator function. $D_t(y|\boldsymbol{x},\boldsymbol{z};\theta_{dt})$ is the target-domain discriminator function.

The source-domain value function is:

$$\begin{aligned}\min_G \max_D V_s(G,D) = {} & \mathbb{E}_{(\boldsymbol{x},\boldsymbol{z})\sim p_s(\boldsymbol{x},\boldsymbol{z})}[\log D_s(\boldsymbol{x},\boldsymbol{z})] \\ & + \mathbb{E}_{\boldsymbol{z}\sim p_s(\boldsymbol{z})}[\log(1 - D_s(G_x(\boldsymbol{z}),\boldsymbol{z}))].\end{aligned} \tag{1}$$

The target-domain value function is:

$$\begin{aligned}\min_G \max_D V_t(G,D) = {} & \mathbb{E}_{\boldsymbol{x}\sim p_t(\boldsymbol{x})}[\log D_t(\boldsymbol{x},G_z(\boldsymbol{x}))] \\ & + \mathbb{E}_{\boldsymbol{z}\sim \widetilde{p}_t(\boldsymbol{z})}[\log(1 - D_t(G_x(\boldsymbol{z}),\boldsymbol{z}))].\end{aligned} \tag{2}$$

In practice, the value functions are reworked such that the generator maximises an inverted expression whose gradient is stronger when the discriminator's output saturates, as in the original GAN paper [6]. Also in practice, the logarithmic functions are replaced with the Wasserstein distance metric which prevents saturation and provides better experimental performance [1]. We start with the above original expressions for the value functions in order to make the following TAN theoretical results a more straightforward extension of the original GAN results.

3.2. Training Procedure

The TAN training procedure is as follows. For m steps, the source-domain value function (eq. 1) is iteratively solved using stochastic gradient descent. Then for n steps, the target-domain value function (eq. 2) is iteratively solved using stochastic gradient descent. The two value functions share a common generator function, $G_x(\boldsymbol{x}|\boldsymbol{z};\theta_{gx})$. This shared generator function learns to accommodate both the source and target domain data, which allows global optimality in the entire TAN framework, as shown in the next section.

Algorithm 1 The TAN Training Procedure

```
$\theta_{gx}, \theta_{gz}, \theta_{ds}, \theta_{dt} \leftarrow$ initialise network parameters
repeat
  for $m$ steps do
    for $k$ steps do
      $(\boldsymbol{x},\boldsymbol{z})^{(1)}, \ldots, (\boldsymbol{x},\boldsymbol{z})^{(M)} \sim p_s(\boldsymbol{x},\boldsymbol{z})$
      $\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(M)} \sim p_s(\boldsymbol{z})$
      $\hat{\boldsymbol{x}}^{(j)} \sim G_x\left(\boldsymbol{z}^{(j)}\right), \quad j = 1, \ldots, M$
      $\rho_r^{(i)} \leftarrow D_s\left((\boldsymbol{x},\boldsymbol{z})^{(i)}\right), \quad i = 1, \ldots, M$
      $\rho_g^{(j)} \leftarrow D_s\left(\hat{\boldsymbol{x}}^{(j)}, \boldsymbol{z}^{(j)}\right), \quad j = 1, \ldots, M$
      $\mathcal{L}_d \leftarrow -\frac{1}{M}\left(\sum_{i=1}^{M}\log\left(\rho_r^{(i)}\right) + \sum_{j=1}^{M}\log\left(1-\rho_g^{(j)}\right)\right)$
      $\theta_{ds} \leftarrow \theta_{ds} - \nabla_{\theta_{ds}}\mathcal{L}_d$
    end for
    $\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(M)} \sim p_s(\boldsymbol{z})$
    $\hat{\boldsymbol{x}}^{(j)} \sim G_x\left(\boldsymbol{z}^{(j)}\right), \quad j = 1, \ldots, M$
    $\rho_g^{(j)} \leftarrow D_s\left(\hat{\boldsymbol{x}}^{(j)}, \boldsymbol{z}^{(j)}\right), \quad j = 1, \ldots, M$
    $\mathcal{L}_g \leftarrow \frac{1}{M}\sum_{j=1}^{M}\log\left(1-\rho_g^{(j)}\right)$
    $\theta_{gx} \leftarrow \theta_{gx} - \nabla_{\theta_{gx}}\mathcal{L}_g$
  end for
  for $n$ steps do
    $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(M)} \sim p_t(\boldsymbol{x})$
    $\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(M)} \sim \widetilde{p}_t(\boldsymbol{z})$
    $\hat{\boldsymbol{z}}^{(i)} \sim G_z\left(\boldsymbol{x}^{(i)}\right), \quad i = 1, \ldots, M$
    $\hat{\boldsymbol{x}}^{(j)} \sim G_x\left(\boldsymbol{z}^{(j)}\right), \quad j = 1, \ldots, M$
    $\rho_e^{(i)} \leftarrow D_t(\boldsymbol{x}^{(i)}, \hat{\boldsymbol{z}}^{(i)}), \quad i = 1, \ldots, M$
    $\rho_g^{(j)} \leftarrow D_t(\hat{\boldsymbol{x}}^{(j)}, \boldsymbol{z}^{(j)}), \quad j = 1, \ldots, M$
    $\mathcal{L}_d \leftarrow -\frac{1}{M}\left(\sum_{i=1}^{M}\log\left(\rho_e^{(i)}\right) + \sum_{j=1}^{M}\log\left(1-\rho_g^{(j)}\right)\right)$
    $\mathcal{L}_g \leftarrow -\frac{1}{M}\left(\sum_{i=1}^{M}\log\left(1-\rho_e^{(i)}\right) + \sum_{j=1}^{M}\log\left(\rho_g^{(j)}\right)\right)$
    $\theta_{dt} \leftarrow \theta_{dt} - \nabla_{\theta_{dt}}\mathcal{L}_d$
    $\theta_{gx} \leftarrow \theta_{gx} - \nabla_{\theta_{gx}}\mathcal{L}_g$
    $\theta_{gz} \leftarrow \theta_{gz} - \nabla_{\theta_{gz}}\mathcal{L}_g$
  end for
until convergence
```

3.3. Global Optimality and Convergence Proof

Proposition 1. *The global optimum across* $\min_G \max_D V_s(G, D)$ *and* $\min_G \max_D V_t(G, D)$ *is achieved at:* $G_z(\boldsymbol{z}|\boldsymbol{x};\theta^*_{gz}) = \frac{p_s(\boldsymbol{x},\boldsymbol{z})\widetilde{p}_t(\boldsymbol{z})}{p_s(\boldsymbol{z})p_t(\boldsymbol{x})}$.

Proof. By straightforward extension of the proof in [6], the following result is achieved on convergence of $\min_G \max_D V_s(G, D)$.

$$p_s(\boldsymbol{x},\boldsymbol{z}) = G_x(\boldsymbol{x}|\boldsymbol{z};\theta^*_{gx})p_s(\boldsymbol{z}). \tag{3}$$

Similarly, by straightforward extension of the proof in [4], the following result is achieved on convergence of $\min_G \max_D V_t(G, D)$.

$$G_x(\boldsymbol{x}|\boldsymbol{z};\theta^*_{gx})\widetilde{p}_t(\boldsymbol{z}) = G_z(\boldsymbol{z}|\boldsymbol{x};\theta^*_{gz})p_t(\boldsymbol{x}). \tag{4}$$

The requirement on convergence from [6] and [4] for $\min_G \max_D V_s(G, D)$ and $\min_G \max_D V_t(G, D)$ is that D is allowed to reach its optimum at each training step, given G. D_s and D_t are each allowed to reach their optimum at each training step of their respective value functions given their respective G. Therefore, $\min_G \max_D V_s(G, D)$ and $\min_G \max_D V_t(G, D)$ will simultaneously converge if $V_s(G, D)$ and $V_t(G, D)$ are convex in G.

Therefore, on simultaneous convergence of the above two value functions,

$$\begin{aligned} G_x(\boldsymbol{x}|\boldsymbol{z};\theta^*_{gx}) &= \frac{G_z(\boldsymbol{z}|\boldsymbol{x};\theta^*_{gz})p_t(\boldsymbol{x})}{\widetilde{p}_t(\boldsymbol{z})} \\ &= \frac{p_s(\boldsymbol{x},\boldsymbol{z})}{p_s(\boldsymbol{z})}. \end{aligned} \tag{5}$$

Finally,

$$G_z(\boldsymbol{z}|\boldsymbol{x};\theta^*_{gz}) = \frac{p_s(\boldsymbol{x},\boldsymbol{z})\widetilde{p}_t(\boldsymbol{z})}{p_s(\boldsymbol{z})p_t(\boldsymbol{x})}. \tag{6}$$

□

As stated in [6], the value function is not expected to be convex w.r.t the generator in practice due to the use of multilayer perceptrons. However, the excellent performance of multilayer perceptrons in practice suggests that they are a reasonable model to use despite their lack of theoretical guarantees.

4. Experiments

We are currently performing extensive experiments and will release the details of these experiments in future versions of this paper.

5. Conclusion

This paper has demonstrated the viability of the TAN framework, suggesting that these research directions could prove useful.

Acknowledgements

Thank you to Sarah Murphy and Lewis Moffat for helpful discussions on the potential applications of TAN.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[2] S. Arora, A. Risteski, and Y. Zhang. Theoretical limitations of encoder-decoder gan architectures. *arXiv preprint arXiv:1711.02651*, 2017.

[3] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *International Conference on Learning Representations2*, 2017.

[4] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. *International Conference on Learning Representations*, 2017.

[5] Z. Gan, L. Chen, W. Wang, Y. Pu, Y. Zhang, H. Liu, C. Li, and L. Carin. Triangle generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 5253–5262, 2017.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[7] Y. Jing, Y. Yang, Z. Feng, J. Ye, and M. Song. Neural style transfer: A review. *arXiv preprint arXiv:1705.04058*, 2017.

[8] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[9] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 4, 2017.