Generative Adversarial Nets for Multiple Text Corpora

1ˢᵗ Abdolghani Ebrahimi
dept. Industrial Engineering
Northwestern University
Evanston, USA
ghani@u.northwestern.edu

2ⁿᵈ Diego Klabjan
dept. Industrial Engineering
Northwestern University
Evanston, USA
d-klabjan@northwestern.edu

3ʳᵈ Baiyang Wang
dept. Industrial Engineering
Northwestern University
Evanston, USA
bwang@northwestern.edu

Abstract—Generative adversarial nets (GANs) have been successfully applied to the artificial generation of image data. In terms of text data, much has been done on the artificial generation of natural language from a single corpus. We consider multiple text corpora as the input data, for which there can be two applications of GANs: (1) the creation of consistent cross-corpus word embeddings given different word embeddings per corpus; (2) the generation of robust bag-of-words document embeddings for each corpora. We demonstrate our GAN models on real-world text data sets from different corpora, and show that embeddings from both models lead to improvements in supervised learning problems.

I. Introduction

Generative adversarial nets (GAN) [1] belong to a class of generative models which are trainable and can generate artificial data examples similar to the existing ones. In a GAN model, there are two sub-models simultaneously trained: a generative model $\mathcal{G}$ from which artificial data examples can be sampled, and a discriminative model $\mathcal{D}$ which classifies real data examples and artificial ones from $\mathcal{G}$. By training $\mathcal{G}$ to maximize its generation power, and training $\mathcal{D}$ to minimize the generation power of $\mathcal{G}$, so that ideally there will be no difference between the true and artificial examples, a minimax problem can be established. The GAN model has been shown to closely replicate a number of image data sets, such as MNIST, Toronto Face Database (TFD), CIFAR-10, SVHN, and ImageNet [1], [2].

The GAN model has been extended to text data in a number of ways. For instance, [3] applied a long-short term memory (LSTM) [4] generator and approximated discretization to generate text data. SeqGAN [5] is another model that mainly addresses two problems faced by GAN models in text generation: difficulty in passing gradient updates to the generative model due to discrete tokens and limitation of discriminative model in assessing partial sequences. Moreover, [6] applied the GAN model to generate dialogues, i.e. pairs of questions and answers. Meanwhile, the GAN model can also be applied to generate bag-of-words embeddings of text data, which focus more on key terms in a text document rather than the original document itself. Such a model is provided by [7] using the energy-based GAN [8]. In addition, GAN is used by [9] along ideas from conditional variational autoencoders to generate plausible paraphrases that conveys the same meaning as the original sentence.

To the best of our knowledge, there has been no literature on applying the GAN model to multiple corpora of text data. Multi-class GANs have been proposed [10; 11], but a class in multi-class classification is not the same as multiple corpora. Because knowing the underlying corpus membership of each text document can provide better information on how the text documents are organized, and documents from the same corpus are expected to share similar topics or key words, considering the membership information can benefit the training of a text model from a supervised perspective. We consider two problems associated with training multi-corpus text data: (1) Given a separate set of word [12], how to obtain a better set of cross-corpus word embeddings from them? (2) How to incorporate the generation of document embeddings from different corpora in a single GAN model?

For the first problem, we train a GAN model which discriminates documents represented by different word embeddings, and train the cross-corpus word embedding so that it is similar to each existing word embedding per corpus. For the second problem, we train a GAN model which considers both cross-corpus and per-corpus "topics" in the generator, and applies a discriminator which considers each original and artificial document corpus. We also show that with sufficient training, the distribution of the artificial document embeddings is equivalent to the original ones. Our work has the following contributions: (1) we extend GANs to multiple corpora of text data, (2) we provide applications of GANs to finetune word embeddings and to create robust document embeddings, and (3) we establish theoretical convergence results of the multi-class GAN model.

Section II reviews existing GAN models related to this paper. Section III describes the GAN models on training cross-corpus word embeddings and generating document embeddings for each corpora, and explains the associated algorithms. Section IV presents the results of the two models on text data sets, and transfers them to supervised learning. Section V summarizes the results and concludes the paper.

II. Literature Review

In a GAN model, we assume that the data examples $\mathbf{x}$ are drawn from a distribution $p_{\mathbf{x}}(\cdot)$, and the artificial data examples

$\mathcal{G}(\mathbf{z}) := \mathcal{G}(\mathbf{z}, \theta_g)$ are transformed from the noise distribution $\mathbf{z} \sim p_{\mathbf{z}}(\cdot)$. The binary classifier $\mathcal{D}(\cdot)$ outputs the probability of a data example (or an artificial one) being an original one. Because the probabilistic structure of a GAN can be unstable to train, the Wasserstein GAN [13] is proposed which applies a 1-Lipschitz function as a discriminator.

We note that in many circumstances, data sets are obtained with supervised labels or categories, which can add explanatory power to unsupervised models such as the GAN. For instance, the CoGAN [10] considers pairs of data examples from different categories, and the weights of the first few layers (i.e. close to $\mathbf{z}$) are tied. The conditional GAN where the generator $\mathcal{G}$ and the discriminator $\mathcal{D}$ depend on the class label y was proposed in [11]. Work [2] applied the class labels for semi-supervised learning with an additional artificial class. However, all these models consider only images and do not produce word or document embeddings, therefore being different from our models.

For generating real text, [3] proposed textGAN in which the generator has an LSTM form, and a uni-dimensional convolutional neural network [14; 15] is applied as the discriminator. Also, a weighted softmax function is applied to make the argmax function differentiable. The focus of our work is to summarize information from longer documents, so we apply document embeddings such as the tf-idf to represent the documents rather than to generate real text.

For generating bag-of-words embeddings of text, [7] proposed a GAN model with the mean squared error of a denoising autoencoder as the discriminator, and the output $\mathbf{x}$ is the one-hot word embedding of a document. Our models are different from this model because we consider tf-idf document embeddings for multiple text corpora in the deGAN model (Section III-B), and weGAN (Section III-A) can be applied to produce word embeddings. Also, we focus on robustness based on several corpora, while [7] assumed a single corpus.

For extracting word embeddings given text data, [12] proposed the word2vec model, for which there are two variations: the continuous bag-of-words (cBoW) model [16], where the neighboring words are used to predict the appearance of each word; the skip-gram model, where each neighboring word is used individually for prediction. In GloVe [17], a bilinear regression model is trained on the log of the word co-occurrence matrix. In these models, the weights associated with each word are used as the embedding. For obtaining document embeddings, the para2vec model [18] adds per-paragraph vectors to train word2vec-type models, so that the vectors can be used as embeddings for each paragraph. A simpler approach by taking the average of the embeddings of each word in a document and output the document embedding is exhibited in [19].

III. Models and Algorithms

Suppose we have a number of different corpora $\mathcal{C}^1, \ldots, \mathcal{C}^M$, which for example can be based on different categories or sentiments of text documents. We suppose that $\mathcal{C}^m = \{d_1^m, \ldots, d_{n_m}^m\}$, $m = 1, \ldots, M$, where each d_i^m represents a document. The words in all corpora are collected in a dictionary, and indexed from 1 to V. We name the GAN model to train cross-corpus word embeddings as "weGAN," where "we" stands for "word embeddings," and the GAN model to generate document embeddings for multiple corpora as "deGAN," where "de" stands for "document embeddings."

A. weGAN: Training cross-corpus word embeddings

We assume that for each corpora $\mathcal{C}^m$, we are given word embeddings for each word $v_1^m, \ldots, v_V^m \in \mathbb{R}^d$, where d is the dimension of each word embedding. We are also given a classification task on documents that is represented by a parametric model $\mathcal{C}$ taking document embeddings as feature vectors. We construct a GAN model which combines different sets of word embeddings $\mathcal{V}^m := \{v_i^m\}_{i=1}^V$, $m = 1, \ldots, M$, into a single set of word embeddings $\mathcal{G} := \{v_i^0\}_{i=1}^V$. Note that $\mathcal{V}^1, \ldots, \mathcal{V}^M$ are given but $\mathcal{G}$ is trained. Here we consider $\mathcal{G}$ as the generator, and the goal of the discriminator is to distinguish documents represented by the original embeddings $\mathcal{V}^1, \ldots, \mathcal{V}^M$ and the same documents represented by the new embeddings $\mathcal{G}$.

Next we describe how the documents are represented by a set of embeddings $\mathcal{V}^1, \ldots, \mathcal{V}^M$ and $\mathcal{G}$. For each document d_i^m, we define its document embedding with $\mathcal{V}^m$ as $g_i^m := f(d_i^m, \mathcal{V}^m)$, where $f(\cdot)$ can be any mapping. Similarly, we define the document embedding of d_i^m with $\mathcal{G}$ as follows, with $\mathcal{G} = \{v_j^0\}_{j=1}^V$ trainable $f_{\mathcal{G}}(d_i^m) := f(d_i^m, \mathcal{G})$. In a typical example, word embeddings would be based on word2vec or GLoVe. Function f can be based on tf-idf, i.e. $f(d_i^m, \mathcal{V}) = \sum_{j=1}^V t_{ij}^m v_j^m$ where v_j^m is the word embedding of the j-th word in the m-th corpus $\mathcal{C}^m$ and $t_i^m = (t_{i1}^m, \ldots, t_{iV}^m)$ is the tf-idf representation of the i-th document d_i^m in the m-th corpus $\mathcal{C}^m$.

To train the GAN model, we consider the following minimax problem

$$\min_{\mathcal{C}, \mathcal{G}} \max_{\mathcal{D}} \sum_{m=1}^M \sum_{i=1}^{n_m} [\log(\mathcal{D}(g_i^m)) + \log(1 - \mathcal{D}(f_{\mathcal{G}}(d_i^m))) \\ - \log(e_{k_i^m}^T \mathcal{C}(f_{\mathcal{G}}(d_i^m)))], \quad (1)$$

where $\mathcal{D}$ is a discriminator of whether a document is original or artificial. Here k_i^m is the label of document d_i^m with respect to classifier $\mathcal{C}$, and $e_{k_i^m}$ is a unit vector with only the k_i^m-th component being one and all other components being zeros. Note that $\log(e_{k_i^m}^T \mathcal{C}(f_{\mathcal{G}}(d_i^m)))$ is equivalent to $KL(e_{k_i^m} \| \mathcal{C}(f_{\mathcal{G}}(d_i^m)))$, but we use the former notation due to its brevity.

The intuition of (1) is explained as follows. First we consider a discriminator $\mathcal{D}$ which is a feedforward neural network (FFNN) with binary outcomes, and classifies the document embeddings $\{f_{\mathcal{G}}(d_i^m)\}_{i=1}^{n_m} {}_{m=1}^M$ against the original document embeddings $\{g_i^m\}_{i=1}^{n_m} {}_{m=1}^M$. Discriminator $\mathcal{D}$ minimizes this classification error, i.e. it maximizes the log-likelihood of $\{f_{\mathcal{G}}(d_i^m)\}_{i=1}^{n_m} {}_{m=1}^M$ having label 0 and $\{g_i^m\}_{i=1}^{n_m} {}_{m=1}^M$ having label 1. This corresponds to

$$\sum_{m=1}^M \sum_{i=1}^{n_m} [\log(\mathcal{D}(g_i^m)) + \log(1 - \mathcal{D}(f_{\mathcal{G}}(d_i^m)))]. \quad (2)$$

For the generator $\mathcal{G}$, we wish to minimize (1) against $\mathcal{G}$ so that we can apply the minimax strategy, and the combined word embeddings $\mathcal{G}$ would resemble each set of word embeddings $\mathcal{V}^1, \ldots, \mathcal{V}^M$. Meanwhile, we also consider classifier $\mathcal{C}$ with K outcomes, and associates d_i^m with label k_i^m, so that the generator $\mathcal{G}$ can learn from the document labeling in a semi-supervised way.

If the classifier $\mathcal{C}$ outputs a K-dimensional softmax probability vector, we minimize the following against $\mathcal{G}$, which corresponds to (1) given $\mathcal{D}$ and $\mathcal{C}$:

$$\sum_{m=1}^{M} \sum_{i=1}^{n_m} \left[\log(1 - \mathcal{D}(f_{\mathcal{G}}(d_i^m))) - \log(e_{k_i^m}^T \mathcal{C}(f_{\mathcal{G}}(d_i^m))) \right]. \quad (3)$$

For the classifier $\mathcal{C}$, we also minimize its negative log-likelihood

$$-\sum_{m=1}^{M} \sum_{i=1}^{n_m} \log(e_{k_i^m}^T \mathcal{C}(f_{\mathcal{G}}(d_i^m))). \quad (4)$$

Assembling (2-4) together, we retrieve the original minimax problem (1).

We train the discriminator and the classifier, $\{\mathcal{D}, \mathcal{C}\}$, and the combined embeddings $\mathcal{G}$ according to (2-4) iteratively for a fixed number of epochs with the stochastic gradient descent algorithm, until the discrimination and classification errors become stable.

Fig. 1 illustrates the weGAN model. The algorithm for weGAN is summarized in Algorithm 1 in the appendix B.

B. deGAN: Generating document embeddings for multi-corpus text data

In this section, our goal is to generate document embeddings which would resemble real document embeddings in each corpus $\mathcal{C}^m$, $m = 1, \ldots, M$. We construct M generators, $\mathcal{G}^1, \ldots, \mathcal{G}^M$ so that $\mathcal{G}^m$ generate artificial examples in corpus $\mathcal{C}^m$. As in Section III-A, there is a certain document embedding such as tf-idf, bag-of-words, or para2vec. Let $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_M\}$. We initialize a noise vector $n = (n_1, \ldots, n_{d_n}) \in \mathbb{R}^{d_n}$, where $n_1, \ldots, n_{d_n} \overset{iid}{\sim} \mathcal{N}$, and $\mathcal{N}$ is any noise distribution.

For a generator $\mathcal{G}^m = \{W_h^m, W_h^0, W_o^m, W_o^0\}$ represented by its parameters, we first map the noise vector n to the hidden layer, which represents different topics. We consider two hidden vectors, h^0 for general topics and h^m for specific topics per corpus, $h^m = a_1(W_h^m n)$, $h^0 = a_1(W_h^0 n)$. Here $a_1(\cdot)$ represents a nonlinear activation function. In this model, the bias term can be ignored in order to prevent the "mode collapse" problem of the generator. Having the hidden vectors, we then map them to the generated document embedding with another activation function $a_2(\cdot)$,

$$o_m = a_2(W_o^m h^m + w_o^0 h^0). \quad (5)$$

To summarize, we may represent the process from noise to the document embedding as

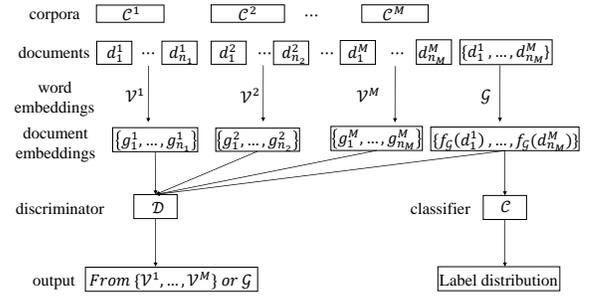$$G^m(n) = a_2(W_o^m a_1(W_h^m n) + w_o^0 a_1(W_h^0 n)). \quad (6)$$

Fig. 1: Model structure of weGAN.

Given the generated document embeddings $\mathcal{G}^1(n), \ldots, \mathcal{G}^M(n)$, we consider the following minimax problem to train the generator $\mathcal{G}$ and the discriminator $\mathcal{D}$:

$$\min_{\mathcal{G}} \sum_{m=1}^{M} E_n \log \left\{ e_{M+m}^T \mathcal{D}_{\mathcal{G}}^*(\mathcal{G}^m(n)) / \right.$$
$$\left. [e_{M+m}^T \mathcal{D}_{\mathcal{G}}^*(\mathcal{G}^m(n)) + e_m^T \mathcal{D}_{\mathcal{G}}^*(\mathcal{G}^m(n))] \right\}, \quad (7)$$

where

$$\mathcal{D}_{\mathcal{G}}^* \in \arg\max_{\mathcal{D}} \sum_{m=1}^{M} E_{d^m \sim p_m} \left[\log(e_m^T \mathcal{D}(d^m)) \right]$$
$$+ \sum_{m=1}^{M} E_n [\log(e_{M+m}^T \mathcal{D}(\mathcal{G}^m(n)))]. \quad (8)$$

Here we assume that any document embedding d^m in corpus $\mathcal{C}^m$ is a sample with respect to the probability density p_m. Note that when $M = 1$, the discriminator part of our model is equivalent to the original GAN model.

To explain (7-8), first we consider the discriminator $\mathcal{D}$. Because there are multiple corpora of text documents, here we consider $2M$ categories as output of $\mathcal{D}$, from which categories $1, \ldots, M$ represent the original corpora $\mathcal{C}^1, \ldots, \mathcal{C}^M$, and categories $M+1, \ldots, 2M$ represent the generated document embeddings (e.g. bag-of-words) from $\mathcal{G}^1, \ldots, \mathcal{G}^M$. Assume discriminator $\mathcal{D}$, a feedforward neural network, outputs the distribution of a text document being in each category. We maximize the log-likelihood of each document being in the correct category against $\mathcal{D}$

$$\sum_{m=1}^{M} E_{p_m} \left[\log(e_m^T \mathcal{D}(d^m)) \right] + \sum_{m=1}^{M} E_n [\log(e_{M+m}^T \mathcal{D}(\mathcal{G}^m(n)))]. \quad (9)$$

Such a classifier does not only classifies text documents into different categories, but also considers M "fake" categories from the generators. When training the generators $\mathcal{G}^1, \ldots, \mathcal{G}^M$, we minimize the following which makes a comparison between the m-th and $(M+m)$-th categories

$$\sum_{m=1}^{M} E_n \log \frac{e_{M+m}^T \mathcal{D}(\mathcal{G}^m(n))}{e_{M+m}^T \mathcal{D}(\mathcal{G}^m(n)) + e_m^T \mathcal{D}(\mathcal{G}^m(n))}. \quad (10)$$

The intuition of (10) is that for each generated document embedding $\mathcal{G}^m(n)$, we need to decrease $e_{M+m}^T \mathcal{D}(\mathcal{G}^m(n))$,
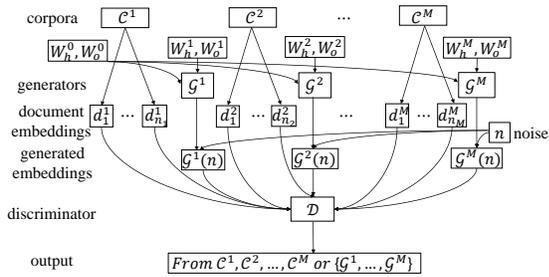
Fig. 2: Model structure of deGAN.

which is the probability of the generated embedding being correctly classified, and increase $e_m^T \mathcal{D}(\mathcal{G}^m(n))$, which is the probability of the generated embedding being classified into the target corpus $\mathcal{C}^m$. The ratio in (10) reflects these two properties.

We iteratively train (9) and (10) until the classification error of $\mathcal{D}$ becomes stable. Fig. 2 illustrates the deGAN model. The algorithm for deGAN algorithm is summarized in Algorithm 2 in the appendix B.

We next show that from (7-8), the distributions of the document embeddings from the optimal $\mathcal{G}^1, \ldots, \mathcal{G}^M$ are equal to the data distributions of $\mathcal{C}^1, \ldots, \mathcal{C}^M$, which is a generalization of [1] to the multi-corpus scenario. The proof of Proposition 1 is in the appendix A.

Proposition 1. Let us assume that the random variables $d^1, \ldots, d^M$ are continuous with probability density $p_1, \ldots, p_M$ which have bounded support $\mathcal{X}$; n is a continuous random variable with bounded support and activations a_1 and a_2 are continuous; and that $\mathcal{G}^{1*}, \ldots, \mathcal{G}^{M*}$ are solutions to (7-8). Then $q_1^*, \ldots, q_M^*$, the probability density of the document embeddings from $\mathcal{G}^{1*}, \ldots, \mathcal{G}^{M*}$, are equal to $p_1, \ldots, p_M$. $\square$

IV. EXPERIMENTS

In the experiments, we consider four data sets, two of them newly created and the remaining two already public: CNN, TIME, 20 Newsgroups (appendix C), and Reuters-21578 (appendix D). The code and the two new data sets are available at github.com/deeplearning2018/emgan. For the pre-processing of all the documents, we transformed all characters to lower case, stemmed the documents, and ran the word2vec model on each corpora to obtain word embeddings with a size of 300. In all subsequent models, we only consider the most frequent 5,000 words across all corpora in a data set.

The document embedding in weGAN is the tf-idf weighted word embedding transformed by the $\tanh$ activation, i.e. $f(d_i^m, \mathcal{V}^m) = \tanh\left(\sum_{j=1}^{V} t_{ij}^m v_j^m\right)$. For deGAN, we use L^1-normalized tf-idf as the document embedding because it is easier to interpret than the transformed embedding using $\tanh$.

For weGAN, the cross-corpus word embeddings are initialized with the word2vec model trained from all documents. For training our models, we apply a learning rate which increases linearly from 0.01 to 1.0 and train the models for 100 epochs with a batch size of 50 per corpus. The classifier $\mathcal{C}$ has a single

hidden layer with 50 hidden nodes, and the discriminator with a single hidden layer $\mathcal{D}$ has 10 hidden nodes. All these parameters have been optimized. For the labels k_i^m in (1), we apply corpus membership of each document.

For the noise distribution $\mathcal{N}$ for deGAN, we apply the uniform distribution $U(-1, 1)$. In (6) for deGAN, $a_1 = \tanh$ and $a_2 = softmax$ so that the model outputs document embedding vectors which are comparable to L^1-normalized tf-idf vectors for each document. For the discriminator $\mathcal{D}$ of deGAN, we apply the word2vec embeddings based on all corpora to initialize its first layer, followed by another hidden layer of 50 nodes. For the discriminator $\mathcal{D}$, we apply a learning rate of 0.1, and for the generator $\mathcal{G}$, we apply a learning rate of 0.001, because the initial training phase of deGAN can be unstable. We also apply a batch size of 50 per corpus. For the softmax layers of deGAN, we initialize them with the log of the topic-word matrix in latent Dirichlet allocation (LDA) [20] in order to provide intuitive estimates.

For weGAN, we consider two metrics for comparing the embeddings trained from weGAN and those trained from all documents: (1) applying the document embeddings to cluster the documents into M clusters with the K-means algorithm, and calculating the Rand index (RI) [21] against the original corpus membership; (2) finetuning the classifier $\mathcal{C}$ and comparing the classification error against an FFNN of the same structure initialized with word2vec (w2v). For deGAN, we compare the performance of finetuning the discriminator of deGAN for document classification, and the performance of the same FFNN. Each supervised model is trained for 500 epochs and the validation data set is used to choose the best epoch.

A. The CNN data set

In the CNN data set, we collected all news links on www.cnn.com in the GDELT 1.0 Event Database from April 1st, 2013 to July 7, 2017. We then collected the news articles from the links, and kept those belonging to the three largest categories: "politics," "world," and "US." We divided these documents into 21,674 training documents, from which 2,708 validation documents are held out, and 5,420 testing documents.

We hypothesize that because weGAN takes into account document labels in a semi-supervised way, the embeddings trained from weGAN can better incorporate the labeling information and therefore, produce document embeddings which are better separated. The results are shown in Table I and averaged over 5 randomized runs. Performing the Welch's t-test, both changes after weGAN training are statistically significant at a 0.05 significance level. Because the Rand index captures matching accuracy, we observe from Table I that weGAN tends to improve both metrics.

Meanwhile, we also wish to observe the spatial structure of the trained embeddings, which can be explored by the synonyms of each word measured by the cosine similarity. On average, the top 10 synonyms of each word differ by 0.22 word after weGAN training, and 20.7% of all words have different top 10 synonyms after training. Therefore, weGAN tends to provide small adjustments rather than structural changes. Table

II lists the 10 most similar terms of three terms, "Obama," "Trump," and "U.S.," before and after weGAN training, ordered by cosine similarity.

We observe from Table II that for "Obama," "Trump" and "Tillerson" are more similar after weGAN training, which means that the structure of the weGAN embeddings can be more up-to-date. For "Trump," we observe that "Clinton" is not among the synonyms before, but is after, which shows that the synonyms after are more relevant. For "U.S.," we observe that after training, "American" replaces "British" in the list of synonyms, which is also more relevant.

We next discuss deGAN. In Table III, we compare the performance of finetuning the discriminator of deGAN for document classification, and the performance of the FFNN initialized with word2vec. The change is also statistically significant at the 0.05 level. From Table III, we observe that deGAN improves the accuracy of supervised learning.

To compare the generated samples from deGAN with the original bag-of-words, we randomly select one record in each original and artificial corpus. The records are represented by the most frequent words sorted by frequency in descending order where the stop words are removed. The bag-of-words embeddings are shown in Table IV.

From Table IV, we observe that the bag-of-words embeddings of the original documents tend to contain more name entities, while those of the artificial deGAN documents tend to be more general. There are many additional examples not shown here with observed artificial bag-of-words embeddings having many name entities such as "Turkey," "ISIS," etc. from generated documents, e.g. "Syria eventually ISIS U.S. details jet aircraft October video extremist..."

We perform dimensional reduction using t-SNE [22], and plot 100 random samples from each original or artificial category. The original samples are shown in red and the generated ones are shown in blue in fig. 3. We do not further distinguish the categories because there is no clear distinction between the three original corpora, "politics," "world," and "US."

We observe that the original and artificial examples are generally mixed together and not well separable, which means that the artificial examples are similar to the original ones. However, we also observe that the artificial samples tend to be more centered and have no outliers (represented by the outermost red oval).

TABLE I: A comparison between word2vec and weGAN in terms of the Rand index and the classification accuracy for the CNN data set.

	w2v-RI	weGAN-RI
mean	67.88%	**68.45%**
sd.	0.02%	0.01%
	w2v-accuracy	weGAN-accuracy
mean	92.05%	**92.36%**
sd.	0.06%	0.03%

TABLE II: Synonyms of "Obama," "Trump," and "U.S." before and after weGAN training for the CNN data set.

| Obama	w2v	Bush Trump Kerry Abe Netanyahu Rouhani Erdogan he Karzai Tillerson
Obama	weGAN	Trump Bush Kerry Abe Netanyahu Erdogan Tillerson he Carter Rouhani
Trump	w2v	Obama Pence Erdogan Bush Duterte he Sanders Macron Christie Tillerson
Trump	weGAN	Obama Pence Bush Christie Sanders Clinton Erdogan Tillerson Macron Duterte
U.S.	w2v	US Pentagon United Iranian NATO Turkish Qatar Iran British UAE
U.S.	weGAN	US Pentagon United Iranian NATO Turkish Iran Qatar American UAE
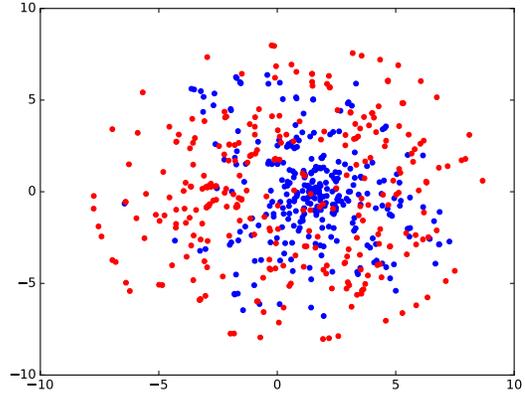
Fig. 3: 2-d representation of original (red) and artificial (blue) examples in the CNN data set.

B. The TIME data set

In the TIME data set, we collected all news links on time.com in the GDELT 1.0 Event Database from April 1st, 2013 to July 7, 2017. We then collected the news articles from the links, and kept those belonging to the five largest categories: "Entertainment," "Ideas," "Politics," "US," and "World." We

TABLE III: A comparison between word2vec and deGAN in terms of the accuracy for the CNN data set.

	w2v-accuracy	deGAN-accuracy
mean	92.05%	**92.29%**
sd.	0.06%	0.09%

TABLE IV: Bag-of-words representations of original and artificial text in the CNN data set.

politics	original	India US Carter defense Indian relationship China said military two
politics	deGAN	year US meeting used read security along building worth foreign
world	original	Turkey Turkish Attack ISIS said Kurdish Erdogan group bomb report
world	deGAN	cut company get lot made code could Steve items may road block phone
US	original	climate change year study according says country temperatures average
US	deGAN	area efforts volunteers town weapons shot local nearly department also

TABLE V: A comparison between word2vec, weGAN, and deGAN in terms of the Rand index and the classification accuracy for the TIME data set.

	w2v-RI	weGAN-RI
mean	70.96%	**71.14%**
sd.	0.02%	0.02%
w2v-accur.	weGAN-accur.	deGAN-accur.
83.79%	84.76%	**85.38%**
0.17%	0.08%	0.11%

divided these documents into 12,286 training documents, from which 1,535 validation documents are held out, and 3,075 testing documents. Table V compares the clustering results of word2vec and weGAN, and the classification accuracy of an FFNN initialized with word2vec, finetuned weGAN, and finetuned deGAN. The results in Table V are the counterparts of Table I and Table III for the TIME data set. The differences are also significant at the 0.05 level.

From Table V, we observe that both GAN models yield improved performance of supervised learning. For weGAN, on an average, the top 10 synonyms of each word differ by 0.27 word after weGAN training, and 24.8% of all words have different top 10 synonyms after training. We also compare the synonyms of the same common words, "Obama," "Trump," and "U.S.," which are listed in Table VI. In the TIME data set, for "Obama," "Reagan" is ranked slightly higher as an American president. For "Trump," "Bush" and "Sanders" are ranked higher as American presidents or candidates. For "U.S.," we note that "Pentagon" is ranked higher after weGAN training, which we think is also reasonable because the term is closely related to the U.S. government. For deGAN, we also compare the original and artificial samples in terms of the highest probability words, which is shown in Table VII. We also perform dimensional reduction using t-SNE for 100 examples per corpus and plot them in Figure 4. All these figures and tables show results similar to Section IV-A.

TABLE VI: Synonyms of "Obama," "Trump," and "U.S." before and after weGAN training for the TIME data set.

Obama	w2v	Trump Bush Xi Erdogan Rouhani Reagan Hollande Duterte Abe Jokowi
Obama	weGAN	Trump Bush Xi Erdogan Reagan Rouhani Hollande Abe Jokowi Duterte
Trump	w2v	Obama Erdogan Rubio Duterte Bush Putin Sanders Xi Macron Pence
Trump	weGAN	Obama Erdogan Rubio Bush Sanders Putin Duterte Xi Macron Pence
U.S.	w2v	NATO Iran Japan Pentagon Russia Pakistan Tehran EU Ukrainian Moscow
U.S.	weGAN	NATO Pentagon Iran Japan Russia Tehran Pakistan EU Ukrainian Moscow

TABLE VII: Bag-of-words representations of original and artificial text in the TIME data set.

Entertainment	original	show London attack people proud open going right according way home
Entertainment	deGAN	music actor Michael John song going meeting James produced pop vocal
Ideas	original	American would service national young country year serve security world
Ideas	deGAN	city project part development grand bear often west new status high agents
Politics	original	Assange embassy BBC Swedish charges told authorities officials Sweden
Politics	deGAN	members present national committee party Paul sign Trump removed brief
US	original	Charleston many Carolina South funeral hand wrote political words
US	deGAN	Davis head board man relationship recent Sunday stone fire wrote gay well
world	original	Erdogan Turkey political power government two leaders minister AKP
world	deGAN	suffering like know old violence local daily young interest three first man

V. Conclusion

In this paper, we have demonstrated the application of the GAN model on text data with multiple corpora. We have shown that the GAN model is not only able to generate images, but also able to refine word embeddings and generate document embeddings. Such models can better learn the inner structure of multi-corpus text data, and benefit supervised learning. The improvements in supervised learning are not large but statistically significant. The weGAN model outperforms deGAN in terms of supervised learning for 3 out of 4 data sets, and is thereby recommended. The synonyms from weGAN
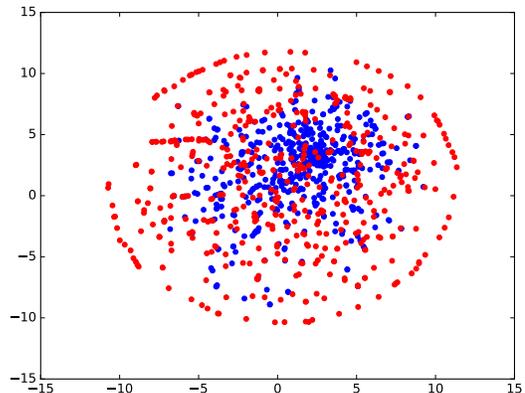

Fig. 4: 2-d representation of original (red) and artificial (blue) examples in the TIME data set.

also tend to be more relevant than the original word2vec model. The t-SNE plots show that our generated document embeddings are similarly distributed as the original ones.

APPENDIX A
PROOF OF PROPOSITION 1

Since $\mathcal{X}$ is bounded, all the integrals exhibited next are well-defined and finite. It follows that for any parameters, $\mathcal{G}^m(n)$ is a continuous random variable with probability density q_m and finite support due to continuity of n, a_1 and a_2. From (8),

$$
\begin{aligned}
\mathcal{D}_{\mathcal{G}}^* = \arg\max_{\mathcal{D}} &\left\{ \sum_{m=1}^{M} \int p_m(x) \log(e_m^T \mathcal{D}(x)) dx \right. \\
&+ \sum_{m=1}^{M} \int q_m(x) \log(e_{M+m}^T \mathcal{D}(x)) dx \Bigg\} \\
= \arg\max_{\mathcal{D}} &\left\{ \int \sum_{m=1}^{M} p_m(x) \log(e_m^T \mathcal{D}(x)) \right. \\
&+ \sum_{m=1}^{M} q_m(x) \log(e_{M+m}^T \mathcal{D}(x)) dx \Bigg\}.
\end{aligned} \tag{11}
$$

This problem reduces to

$$
\max_{b_1,\ldots,b_m} \sum_{m=1}^{M} a_m \log b_m \text{ subject to } \sum_{m=1}^{M} b_m = 1, \tag{12}
$$

the solution of which is $b_m^* = a_m / \sum_{m=1}^{M} a_m$, $m = 1, \ldots, M$. Therefore, the solution to (11) is

$$
\mathcal{D}_{\mathcal{G}}^*(x) = \frac{(p_1(x),\ldots,p_M(x),q_1(x),\ldots,q_M(x))}{\sum_{m=1}^{M} p_m(x) + \sum_{m=1}^{M} q_m(x)}. \tag{13}
$$

We then obtain from (7) that $q_1^*,\ldots,q_M^*$, are solutions to the following:

$$
\begin{aligned}
\min_{q_1,\ldots,q_M} &\sum_{m=1}^{M} \int q_m(x) \cdot \log\left[\frac{q_m(x)}{q_m(x) + p_m(x)}\right] dx \\
= \min_{q_1,\ldots,q_M} &\; c + \sum_{m=1}^{M} \int q_m(x) \log\left[\frac{q_m(x)}{(q_m(x) + p_m(x))/2}\right] dx \\
= \min_{q_1,\ldots,q_M} &\; c + \sum_{m=1}^{M} KL(q_m \| (q_m + p_m)/2),
\end{aligned}
$$

where $c = -M \log 2$ is a constant. From non-negativity of the Kullback-Leibler divergence, we conclude that

$$
(q_1^*,\ldots,q_M^*) = (p_1,\ldots,p_M). \qquad \square
$$

APPENDIX B
ALGORITHMS FOR WEGAN AND DEGAN

Algorithm 1 Algorithm for weGAN

1) Train $\mathcal{G}$ based on f from all corpora $\mathcal{C}^1,\ldots,\mathcal{C}^M$.
2) Randomly initialize the weights and biases of the classifier $\mathcal{C}$ and discriminator $\mathcal{D}$.
3) Until maximum number of iterations reached
   a) Update $\mathcal{C}$ and $\mathcal{D}$ according to (2) and (4) given a mini-batch S_1 of training examples $\{d_i^m\}_{i,m}$.
   b) Update $\mathcal{G}$ according to (3) given a mini-batch S_2 of training examples $\{d_i^m\}_{i,m}$.
4) Output $\mathcal{G}$ as the cross-corpus word embeddings.

Algorithm 2 Algorithm for deGAN

1) Randomly initialize the weights of $\mathcal{G}^1,\ldots,\mathcal{G}^M$.
2) Initialize the discriminator $\mathcal{D}$ with the weights of the first layer (which takes document embeddings as the input) initialized by word embeddings, and other parameters randomly initialized.
3) Until maximum number of iterations reached
   a) Update $\mathcal{D}$ according to (9) given a mini-batch of training examples d_i^m and samples from noise n.
   b) Update $\mathcal{G}^1,\ldots,\mathcal{G}^M$ according to (10) given a mini-batch of training examples d_i^m and samples form noise n.
4) Output $\mathcal{G}^1,\ldots,\mathcal{G}^M$ as generators of document embeddings and $\mathcal{D}$ as a corpus classifier.

APPENDIX C
THE 20 NEWSGROUPS DATA SET

The 20 Newsgroups data set is a collection of news documents with 20 categories. To reduce the number of categories so that the GAN models are more compact and have more samples per corpus, we grouped the documents into 6 super-categories: "religion," "computer," "cars," "sport," "science," and "politics" ("misc" is ignored because of its noisiness). We considered each super-category as a different corpora. We then divided these documents into 10,708 training documents, from which 1,335 validation documents are held out, and 7,134 testing documents. We train weGAN and deGAN in the the beginning of Section IV, except that we use a learning rate of 0.01 for the discriminator in deGAN to stabilize the cost function. Table VIII compares the clustering results of word2vec and weGAN, and the classification accuracy of the FFNN initialized with word2vec, finetuned weGAN, and finetuned deGAN. All comparisons are statistically significant at the 0.05 level. The other results are similar to the previous two data sets and are thereby omitted here.

APPENDIX D
THE REUTERS-21578 DATA SET

The Reuters-21578 data set is a collection of newswire articles. Because the data set is highly skewed, we considered the eight categories with more than 100 training documents:

TABLE VIII: A comparison between word2vec, weGAN, and deGAN in terms of the Rand index and the classification accuracy for the 20 Newsgroups data set.

height	w2v-RI	weGAN-RI
mean	76.14%	**76.74%**
sd.	0.02%	0.08%
w2v-accur.	weGAN-accur.	deGAN-accur.
87.34%	**89.90%**	89.32%
0.04%	0.02%	0.15%

"earn," "acq," "crude," "trade," "money-fx," "interest," "money-supply," and "ship." We then divided these documents into 5,497 training documents, from which 692 validation documents are held out, and 2,207 testing documents. We train weGAN and deGAN in the same way as in the 20 Newsgroups data set. Table IX compares the clustering results of word2vec and weGAN, and the classification accuracy of the FFNN initialized with word2vec, finetuned weGAN, and finetuned deGAN. All comparisons are statistically significant at the 0.05 level except the Rand index. The other results are similar to the CNN and TIME data sets and are thereby omitted here.

TABLE IX: A comparison between word2vec, weGAN, and deGAN in terms of the Rand index and the classification accuracy for the Reuters-21578 data set.

height	w2v-RI	weGAN-RI
mean	71.28%	**71.43%**
sd.	0.26%	0.07%
w2v-accur.	weGAN-accur.	deGAN-accur.
92.86%	**95.10%**	94.86%
0.09%	0.10%	0.10%

REFERENCES

[1] 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 27*, 2014.

[2] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, 2016.

[3] Y. Zhang, Z. Gan, and L. Carin, "Generating text via adversarial training," in *Workshop on Adversarial Training (NIPS 2016)*, 2016.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[6] J. Li, W. Monroe, T. Shi, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," Tech. Rep., 2017, arXiv:1701.06547.

[7] J. Glover, "Modeling documents with generative adversarial networks," in *Workshop on Adversarial Training (NIPS 2016)*, 2016.

[8] J. J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial networks," in *5th International Conference on Learning embeddings (ICLR 2017)*, 2017.

[9] Q. Yang, D. Shen, Y. Cheng, W. Wang, G. Wang, L. Carin *et al.*, "An end-to-end generative architecture for paraphrase generation," 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)*, 2019, pp. 3123–3133.

[10] M. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, 2016.

[11] M. Mirza and S. Osindero, "Conditional generative adversarial nets," Tech. Rep., 2014, arXiv:1411.1784.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, Lake Tahoe, Nevada, USA, 2013.

[13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," Tech. Rep., 2017, arXiv:1701.07875.

[14] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural language processing (almost) from scratch." *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[15] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Workshop (ICLR 2013)*, 2013.

[17] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014.

[18] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, 2014.

[19] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, 2013.

[20] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[21] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, pp. 846–850, 1971.

[22] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.