TOPIC AWARE NEURAL LANGUAGE MODEL: DOMAIN ADAPTATION OF UNCONDITIONAL TEXT GENERATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Our goal is to adapt pre-trained neural language models (NLMs) to the unconditional text generation task within the target domain. Because many Transformer based NLMs are trained on more massive and heterogeneous corpora than this target corpus, the difference between these corpora and this domain corpus raises the question of whether these NLMs can provide their benefits to this task even after the fine-tuning. To tackle this problem, our approach focuses on topics to bridge the semantic gap between these corpora and the target domain corpus, and injects topics into Transformer. This approach develops a topic alignment (TA) as manipulation, and Topic Distribution Modeling (TDM) and Topic Embedding Modeling (TEM) as training tasks. Experiments show that our model contributes to resolve this imbalance problem, and can tailor pre-trained NLMs to generate coherent and semantically valid text reflecting a given small fine-tuning corpus.

1 INTRODUCTION

Our goal is to adapt pre-trained neural language models (NLMs) to the unconditional text generation task for the target domain. As with global semantic information, topic models (Blei et al., 2003; Kawamae, 2018; Wang et al., 2020) and their extensions take a global statistical view and look at the word distributions of topics across a given corpus; they represent each document as a bag-of-word (BOW) vector. Topics are global variables, describes the distributions over all tokens in the vocabulary, and form interpretable representations over texts. Although these models organize a given corpus into small sets of prominent topics and have been proven to be powerful tools for uncovering latent structure, they are not, in the strict sense, sequence models. Recently, Transformer has been applied as pre-trained NLMs (Vaswani et al., 2017; Yang et al., 2019), and are shown to be highly successful in the domain of natural language processing.

Their success motivate us to explore how to adapt Transformer based NLMs for reflecting a given fine-tuning data and generating more private text with a few modifications than ever before. The corpus size of this fine-tuning is generally smaller than that of pre-training corpora, and may be unbalanced for training NLMs. For example, the size of the popular pre-training data, Giga5en (Parker et al., 2011), and ClueWeb 2012-B[1] is 16G, and 25TB, respectively. The fine-tuning needs a different approach to training NLMs than the pre-training approach, as pre-trained NLMs should be biased towards the patterns of language used in the training data (Keskar et al., 2019), and may not lead to domain adaptation.

To tackle this imbalance problem, we explore how to adapt pre-trained Transformer based NLMs to the target domain in the fine-tuning, while preserving their advantages. Since topics exist over these datasets and represents their semantic structures such as higher-order and non-linear interaction between words, our model, Topic Aware Neural Language Model (TAN), focuses on topics to resolve this imbalance between these corpora and bridge their semantic gap. As topics reflect a set of co-occurring words, semantic information, and their

[1] `https://www.lemurproject.org/clueweb09.php/`

syntactic structure, as global statistical information, TAN introduces a new manipulation, "topic-alignment (TA)", and new training tasks such as Topic Distribution Modeling (TDM) and Topic Embedding Modeling (TEM) While previous Transformer based NLMs are better at learning from the predefined segment length such as the context, local information using Multi-head attention (Vaswani et al., 2017), TAN injects topics into Transformer to train NLMs by fine-tuning with emphasizing these dependencies over segments in the target domain via topics. This model not only captures global semantic coherence of all segments and word concurrence patterns, but also enriches the representation of each token by adapting it to its local context, which goes beyond the segment it resides in and can be flexibly defined according to the target task.

Experiments confirm that TAN and its augmentations supports existing state-of-the-art NLMs and verify its following advantages;
•Theoretical contributions: TAN adapts Transformer based pre-trained NLMs to the unconditional text generation task via topics while preserving their architectuures.
•Practical contributions: As TAN is a flexible plug-and-play model, and does not need to update the parameters of pre-trained Transformer NLMs, it generates more target-specific text at a lower computational cost than using previous NLMs alone.

2 PREVIOUS WORK

Recently, pre-trained neural language models (NLMs), such as BERT (Devlin et al., 2019), GPT2 (Radford et al., 2019), XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2020) use of Transformer (Vaswani et al., 2017) for learning contextualized text representations, and have yielded great advances for NLP tasks. These NLMs can be fine-tuned on many natural language understanding (NLU) tasks such as named entity recognition, question answering and text classification.

Though achieving appealing performances, these Transformer-based models are better at exploring the relationships among local tokens than the document global semantics (Wang et al., 2020). As no Transformer-based model considers these explicit semantics, Wang et al (Wang et al., 2020) rearrange and further explore the semantics of the topic model and develop a friendly topic assistant for Transformer-based abstractive summarization models. UNIfied pre-trained Language Model (UNILM) (Dong et al., 2019) is applied to NLU and natural language generation (NLG) tasks by employing a shared Transformer network and utilizing specific self-attention masks to control what context the prediction is conditioned on. However, Transformers are usually trained on disjoint fixed-length segments, without any information flow across segments (Dai et al., 2019), limiting the contextualization within the current segment. Therefore, they often fail to take full advantage of many other rich contextual information, such as longer-range word dependencies beyond the segment length and semantic relationships between neighboring segments. While BertSUM (Wang et al., 2020) notes that topic models are better at learning explicit document semantics than Transformers, our model is applied to the domain adaptation of text generation task, and has the different architecture and training tasks to perform topic transfer between domains.

3 Methodology

3.1 Problem formulation

Our approach aims to adapt pre-trained NLMs so that they generate unconditional texts reflecting the target-domain corpus more than corpora used in the pre-training.

In NLP, language models are trained as conditional language models for specific tasks that require text generation Bengio et al. (2003). Given text sequence $\mathbf{x} = \{x_1, \cdots, x_T\}$ and a dataset $D = \{\mathbf{x}^1, \cdots, \mathbf{x}^D\}$, NLMs are pre-trained by minimizing the negative likelihood under forward autoregressive factorization:

$$min_\theta \mathcal{L}_{LM}(\theta) = min_\theta - \sum_{d=1}^{|D|} \sum_{t=1}^{T} \log P_\theta(x_{d,t} | \mathbf{x}_{d,1:t-1}), \tag{1}$$

where θ represents model parameters.

Since our approach focuses on the fact that the gap between the corpora for pre-trained models and the corpus of target domain is different in this distribution, it aims to train these models by bringing these distributions to the distributions observed in the target domain. For example, given "My favorite artist is", the pre-trained model predicts "Michelangelo" as the next word, while the fine-tuned model predicts "Botticelli". This leads us to introduce topics, z, into the NLMs and then modify Eq (1) to:

$$min_\theta \mathcal{L}_{TLN}(\theta) = min_\theta - \sum_{d=1}^{|D|} \sum_{t=1}^{T} \log \sum_{z_t=1}^{Z} P_\theta(x_{d,t}|z_t) P_\theta(z_t|\mathbf{x}_{d,1:t-1}), \tag{2}$$

where z_t denotes the topic of t-th token, and Z is the number of topics. Different from the previous NLMs, TAN explicitly introduces topics into the generative process to provide richer contextual information for improving NLMs. This paper explores how to discover these topics, and fit this obtained these topics to existing pre-trained NLMs.

3.2 Our motivation and Architecture design

Our motivation for adaptation is to discover topics from a target domain and teach them to pre-trained NLMs while preserving the semantic meaning and language structural information that these NLMs have. While Transformer encodes context, as local information, it requires the large size of source target data set to learn the higher-order and non-linear interaction between words, and will need more parameters, computation resource and time. Further, due to the limited position index during pre-training, most Transformer-based models have a maximum capacity of input tokens. Thus, they often truncate the length of a document to satisfy the length limitation of the encoder, which may lose some important semantics, especially for long documents. It is often observed that the learned attentive patterns of many heads are not as reasonable as we expect (Michel et al., 2019), and we might obtain this global information in the upper blocks by increasing the number of blocks in transformers (Dosovitskiy et al., 2021), where the transformer architecture requires a large number of parameters and its computational cost is very high. This motivates us to discover explicitly topics from a target domain share them with pre-trained NLMs.

As 1) a topic describes a co-occurrence pattern of tokens with similar semantics, and 2) the differences between pre-training and fine-tuning data sets are not only in the topic itself, but also in the ratio of topics, our model needs to be trained separately from pre-trained NLMs so that topics, $\mathbf{z}$, are not buried. This leads our model to have both the encoder and the decoder, as shown in Figure 1. The encoder discovers topics from a given target domain and fuses topics to the decoder with their distributions. The difference between this encoder and the other encoders is in finding topic distributions from a given target domain, aligning topics between the encoder and the decoder on the top level encoder/decoder block, and replacing the word generation network in the decoder with these topic distributions. The decoder can reuse pre-trained NLMs (e.g., GPT2), jointly learns and shares topics with the encoder. Therefore, this model does not break any structure of the original Transformer network, and can preserve the power of pre-trained NLMs.

3.3 Input

Given a target domain corpus, TAN feeds the same text as input to the encoder (the source text), and the decoder each (the target text), from each of this target, as shown in Fig 1 It layers convert the inputs into token (linguistic) embedding, and adds special tokens [CLS], [SEP], [EOS], and <s>. Following the text preprocessing of other Transformer based NLMs, TAN tokenizes each input text as the linguistic input of token embedding, where each subword is embedded with Word Piece (Wu et al., 2016) or other model-specific tokenizer (e.g., Byte-Pair Encoding (BPE) vocabulary (Radford et al., 2019)) whose length equals the length of its input. [CLS] token is only inserted prior to the token, and denotes the class of each source text. [SEP] token is assigned to the end of each sentence in each input sequence, and indicates a sentence break. [EOS] token is assigned only after the last token in each input sequence. <s> token is only inserted prior to the token in each target text. A learnable

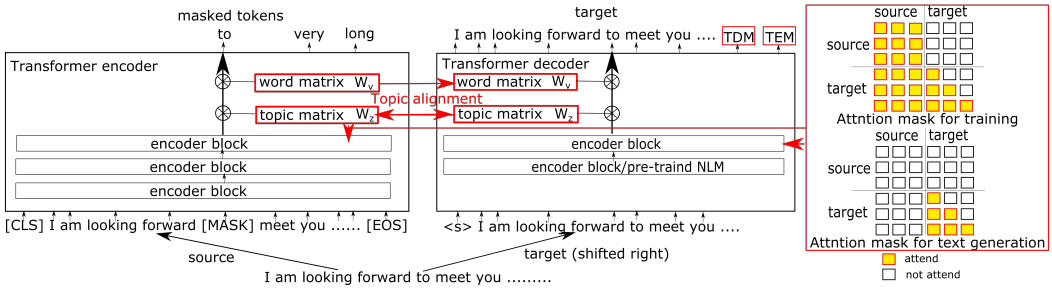

Figure 1: TAN consists of (left) the encoder and the decoder, and (right) two setting masks: The areas highlighted in red are the newly introduced elements. In the training phase, TAN accepts the same input sequence as the source text, and the target text, learns W_Z and W_V, and makes topics share with the encoder and the decoder via topic alignment, attention mask for training, TDM and TEM. In the generation phase, TAN uses only the decoder (pre-trained NLM) using the attention mask for text generation, W_Z, and W_V.

sequence position embedding is added to every input element indicating its order in the input sequence, the same as other models. Because the Transformer encoder has no recurrence, it adds some information about positions into the input embeddings. As the positions of [CLS], [SEP] and [EOS], are fixed in every input sequence, their position embeddings are also fixed as constant values for every input sequence. The final representation for each token is obtained by summing these embedding.

3.4 Attention mechanism with two setting masks

In order to effectively discover and fuse topics in a unified manner, TAN integrates both the encoder and decoder when fine-tuning on a given corpus. This idea leads us to introduce two different self attention mask (e.g., the bidirectional mask and the left-to-right mask) to seamlessly support both the encoder and decoder, and apply it only to the top layer block on behalf of the conventional mask, as shown in Fig 1. We denote the embedded inputs as $\mathbf{H}^0 = [e_1, \cdots, e_{|X|}]$ and then encode them into multiple levels of contextual representations $\mathbf{H}^l = [h_1^l, \cdots, h_{|X|}^l]$ using L-stacked Transformer blocks, where the l-th Transformer block is denoted as $\mathbf{H}^l = Transformer(\mathbf{H}^{l-1}), l \in [1, L]$. Inside each Transformer block, the previous layer's output $\mathbf{H}^{l-1} \in \mathbb{R}^{|x| \times d_h}$ is aggregated using multi-head self-attention, and the core of block is multi-head attention with heads that uses a causal mask to preclude attending to future tokens using the scaled dot-product attention:

$$
\begin{aligned}
&\boldsymbol{Q} = \mathbf{H}^{l-1}\boldsymbol{W}_l^Q, \boldsymbol{K} = \mathbf{H}^{l-1}\boldsymbol{W}_l^K, \boldsymbol{V} = \mathbf{H}^{l-1}\boldsymbol{W}_l^V, \\
&Attention(Q, K, V) = softmax(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}} + \boldsymbol{M})\boldsymbol{V}, \\
&\mathbf{M}_{ij} = \begin{cases} -\infty & \text{if } j \text{ is the linguistic token in the decoder and } i < j, \\ 0 & \text{else} \end{cases},
\end{aligned}
\tag{3}
$$

where $\boldsymbol{W}_l^Q, \boldsymbol{W}_l^K, \boldsymbol{W}_l^V \in \mathbb{R}^{d_h \times d_k} \in \mathbb{R}^{d_h \times d_k}$ are learnable weights for computing the queries, keys, and values, $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{x \times d_k}$, respectively, d is the shared dimensionality of the queries and keys. The self-attention mask, $\boldsymbol{M} \in \mathbb{R}^{x \times x}$, determines whether a position can attend to other positions, where, $M_{ij} = 0$ allows the i-th position to attend to the j-th position and $M_{ij} = -\infty$ prevents it from attending. While the encoder allows each linguistic tokens to attend all other tokens in the input text (source), and the decoder use the mask that restricts each linguistic token to attend to only linguistic tokens in the previous position over the same text (left-to-right). As TAN is designed to adopt pre-trained NLMs as the decoder, it extends this mask to attend to all linguistic tokens in the encoder for topic alignment, the attention mask for training, and applies on the top of the top Transformer block. Then, $Attention(Q, K, V)$ is passed to a feedforward layer to compute $\mathbf{H}^l$ for the next layer.

3.5 Topic alignment (TA) and topic/word matrix

As shown in Eq (2), the decoder generates an output sequence of symbols, one element at a time, where $P_\theta(z_t|\mathbf{x}_{d,1:t-1})$ is gained from the decoder (pre-trained NLMs). To adapt pre-trained NLMs to the target domain, Topic alignment (TA) aims to make the encoder assign $P_\theta(x_{d,t}|z_t)$ to the corresponding topic in $P_\theta(z_t|\mathbf{x}_{d,1:t-1})$, where $P_\theta(x_{d,t}|z_t)$ is gained from the target domain corpus if this topic is in this corpus.

While the core of the first block is multi-head attention with k-heads, the core of the second block is a feedforward network with ReLU activation (Nair & Hinton, 2010) that projects inputs to an inner dimension f, with layer normalization (Ba et al., 2016). Using Eq (3), the output l-th layer, X_l,

$$\begin{aligned}
&\bar{X}_l = LayerNorm(X_l), \quad H_l = MultiHead(\bar{X}_l) + \bar{X}_l, \\
&\bar{H}_l = LayerNorm(H_l), \quad X_{l+1} = FFN(\bar{H}_l) + \bar{H}_l, \quad FFN(\bar{H}_l) = max(0, \bar{H}_l U)V, \\
&Multihead(Q, K, V) = [h_1; \cdots; h_k]\boldsymbol{W}_o \quad where h_j = Attention(Q_j, K_j, V_j),
\end{aligned} \tag{4}$$

where $\boldsymbol{U}, \boldsymbol{V} \in \mathcal{R}^{d\times f}$, and $\boldsymbol{W}_o \in \mathbb{R}^{kd_h\times d_k}$ are learnable weights, $\boldsymbol{Q}_j, \boldsymbol{K}_l, \boldsymbol{V}_j \in \mathbb{R}^{x\times d_k}$ are obtained by transforming $\mathbf{H}^l$ using $\boldsymbol{W}_l^{Qj}, \boldsymbol{W}_l^{Kj}, \boldsymbol{W}_l^{Vj} \in \mathbb{R}^{d_h\times d_k}$, respectively. The output of the final pointwise feed-forward layer goes through a final linear layer that acts as a classifier.

As TAN injects the topic on the top blocked attention layers, it maps the hidden representation vector $\boldsymbol{h} \in \mathbb{R}^d$ into a latent topic vector $\boldsymbol{z} \in \mathbb{R}^K$, and then projects this topic vector into the topic specific distribution over words, where K is the topic size. This leads us to define topic matrix, $\boldsymbol{W}_Z \in \mathbb{R}^{d\times Z}$, and word matrix, $\boldsymbol{W}_V \in \mathbb{R}^{Z\times V}$, where V is the size of vocabulary, and adopts this method to sample the next token according to the probability:

$$\mathcal{X} = LayerNorm(X_L)\boldsymbol{W}_Z\boldsymbol{W}_V, \tag{5}$$

where $\boldsymbol{W}_Z$ and $\boldsymbol{W}_V$ are learnable weight.

Note that just as Eq (1) is transformed into Eq (2) with the introduction of topics, $\boldsymbol{W} \in \mathbb{R}^{d\times V}$ used in previous Transformer based models is decomposed into the product of $\boldsymbol{W}_Z$ and $\boldsymbol{W}_V$ in Eq (5). Different from the other Transformer based NLMs, TAN 1) aligns the t+1-th topic of source text with the t-th topic of target text, $P_\theta(z_t|\mathbf{x}_{d,1:t-1})$, and shares these topic specific distributions over words, $P_\theta(x_{d,t}|z_t)$, between the encoder and the decoder, 2) applies linguistic tokens to left-to-right attention on the input sequence including the encoded tokens using the attention mask for training.

To perform TA efficiently, TAN allows the encoder to share some topics, $\mathbb{K} \in \mathbb{Z}$, with the decoder, refer to $\boldsymbol{w}_k \in \boldsymbol{W}_Z$ and update $\boldsymbol{w}_k \in \boldsymbol{W}_v$ on $k \in \mathbb{K}$, and makes the decoder update $\boldsymbol{w}_k \in \boldsymbol{W}_V$ for $k \in \mathbb{Z}\backslash\mathbb{K}$. As a result, TAN can update the parameters about the topics, $\boldsymbol{W}_Z$ and $\boldsymbol{W}_V$, that appear in the target domain, while maintaining the structure and other parameters of the pre-trained NLMs.

Then, the next token is chosen by sampling on a multinomial distribution with these probabilities clipped at the top-k tokens. As temperature-controlled stochastic sampling methods are used for generating text from a trained NLMs, it allows TAN to sample the next token according to the probability:

$$p(w_i) = \frac{exp(x_i/T)}{\sum exp(x_i/T)}, \tag{6}$$

where $T > 0$ is temperature and $x_i \in \mathcal{X}$ is the score of the i-th token in the vocabulary, as shown in Eq (5). Then, the next token is chosen by sampling on a multinomial distribution with these probabilities clipped at the top-k tokens. While $T \to 0$ approximates a greedy distribution, which magnifies the peaks in the probability distribution, $T \to \infty$ flattens the distribution and makes it more uniform.

4 Model Training

4.1 Training objectives of TAN

Our training consists of three tasks: Masked LM (MLM) (Vaswani et al., 2017), and newly introduced Topic Distribution Modeling (TDM) and Topic Embedding Modeling (TEM). Like topics, TDM and TEM are cooperated with some basic Transformer modules, such as embedding and multi-head attention, while MLM has been employed in the encoder. Therefore, we can plug an arbitrary combination of these two modules into various Transformer-based models.

Topic Distribution Modeling (TDM): The objective of TDM is to minimize the difference between a document specific topic distribution and an estimated topic distribution, where the former distribution is the softmax of average over hidden vectors, $\bar{\boldsymbol{w}} \in \mathbb{R}^K$ in each input text, and the latter distribution is the softmax of average over observed tokens, $\hat{\boldsymbol{w}} \in \mathbb{R}^K$, in each input text. It is often observed that the learned attentive patterns of many heads are not as reasonable as we expect (Michel et al., 2019). This motivates us to employ the semantic "distribution over topics" as a token representation to construct an explicit semantic-similarity matrix among tokens, which is further used as the attention weights of a newly added head. The topic-proportion vector is a low-dimensional text representation, conditioned on which we infer a document-related bias to modulate some hidden layers of the encoder/decoder. Among L2 regression, cross-entropy, and KL-Divergence, our experiments confirm that TDM with KL-Divergence, $\mathcal{L}_{TDM}(\theta)$, attains better performance than the others, and so add it to our framework. The distance between the document specific topic distribution, softmax($\bar{\mathbf{w}}_d$), and the topic distribution gained from tokens in each document, softmax($\hat{\mathbf{w}}_d$) defined as:

$$\mathcal{L}_{TDM}(\theta) = \sum_{d=1}^{D} (\text{softmax}(\bar{\boldsymbol{w}}_d) \parallel \text{softmax}(\hat{\boldsymbol{w}}_d)), \tag{7}$$

where $\bar{\boldsymbol{w}}_d$ is the average over hidden vectors of d-th input text, and $\hat{\boldsymbol{w}}_d$ is the weight over topics from observed tokens in d-th text.

Topic Embedding Modeling (TEM): The objective of TEM is the same as TDM, but it differs in using the embedding representation instead of the topic representation. While each linguistic has the embedding representation, $\boldsymbol{E}_v \in \mathbb{R}^{V \times E}$, gained in the first layer of decoder (pre-trained NLMs) and encoder, TAN introduces a learnable topic-embedding matrix, $\boldsymbol{E}_k \in \mathbb{E}^{Z \times E}$ to gain embedding representations through topics. Among L2 regression, cross-entropy, and KL-Divergence, our experiments confirm that TEM with cross-entropy, $\mathcal{L}_{TEM}(\theta)$, attains better performance than the others. The distance between the mean of $\boldsymbol{E}_v$ over observed tokens in d-th text, $\mathbf{e}_{v,d}$, and the mean of $\boldsymbol{w}_k \boldsymbol{E}_e$ over observed hidden vector in d-th text, $\mathbf{e}_{k,d}$ defined as:

$$\mathcal{L}_{TEM}(\theta) = \sum_{d=1}^{D} D_{CE}(\boldsymbol{e}_{v,d} \parallel \boldsymbol{e}_{k,d}). \tag{8}$$

4.2 Fine-tuning of TAN

Once pre-trained, the gained parameters, θ, of Eq (1) is used to initialize the decoder of TAN, and a fine-tuning process is employed to adapt θ to the fine-tuning data. To optimize these parameters and bridge the gap between the data used in the pre-training and the fine-tuning process, we optimize the model loss in this tuning process. Using Eq (2),(7),(8), and $\mathcal{L}_{MLM}(\theta)$ (Vaswani et al., 2017), we can define the loss function, $\mathcal{L}(\theta)$, as the sum of these objective functions to be optimized in the fine-tuning stage:

$$\mathcal{L}_{TAN}(\theta) = \underbrace{\mathcal{L}_{TLM}(\theta) + \lambda_{TDM}\mathcal{L}_{TDM}(\theta) + \lambda_{TEM}\mathcal{L}_{TEM}(\theta)}_{decoder} + \underbrace{\lambda_{MLM}\mathcal{L}_{MLM}(\theta)}_{encoder}, \tag{9}$$

where θ is the parameter set of TAN, λ_{TDM}, λ_{TEM}, and λ_{MLM} are hyper parameters to balance the importance TDM, TEM and MLM. As with parameter update, we use Adaptive Moment Estimation (Adam) (Kingma & Ba, 2015) over mini-batches, and adopt the dropout strategy (Srivastava et al., 2014) to optimize networks.

Table 1: Basic statistics of the datasets in this paper: In the attributes, $\#Z$ denotes the number of topics and $\#K$ is the number for the encoder to refer to in $\#Z$.

Dataset	#reviews	#vocabulary	#Z	#K
Amazon	210,000	246,534	110	100
Yelp	6,685,900	365,762	220	200

5 EXPERIMENTS

5.1 Datasets and Experiment design

Datasets We conducted evaluations using the Amazon review data sets[2] and Yelp[3], as they are large publicly available datasets. Each record in the dataset contains a review text, review title, star rating, anonymized reviewer ID, anonymized product ID and coarse-grained product category, we use only review texts for fine-tuning. All reviews are truncated after 2,000 characters, and all reviews are at least 50 characters long. Of these languages, we use only English for ease of interpretation of the results. We apply the same pre-treatment to Yelp data set and statistics of the resulting data set are shown in Table 1. We used 90%, 5%, and 5% of each data set as training, validation and test sets. The final performance comparison results are derived from the test set.

Experiment Setup We implement our model using Pytorch 1.7.1[4] and will release these codes later. In Eq (9), we set λ_{TDM}, λ_{TEM}, and λ_{MLM} to 0.5, 0.5, and 1, respectively. As TAN uses GPT2 as the pre-trained NLM, it has a stack of 6(encoder)/24(decoder) Transformer blocks with 1024 hidden size was adopted , with 12 attention heads makes the encoder have the same tokenizer with GPT2. where the maximum length of input sequence was set to 64, and the weight matrix of the softmax classifier was tied to token embeddings. As with training setting, we use Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$ is used for optimization, over mini-batches to update parameters and the dropout strategy (Srivastava et al., 2014) was adopted for network optimization. The learning rate is 3e-5, with linear warmup over the first 500 steps and linear decay, where we set the dropout rate, the weight decay, and the batch size to 0.1, 0.01, and 256, respectively. We fine-train all models on 8 Nvidia Tesla V100 GPUs with 32G memory.

5.2 Topic Coherence

The aim of this experiment is to evaluate how well TAN discovers topics, and compare TAN with existing topic based embedding models. In order to assess their quality quantitatively, we use the topic coherence measure (Mimno et al., 2010) to assess the relatedness of the top-ranked words. This measure matches the intuition that top ranked words for the same topic tend to co-occur in documents that address the same topic and all are closely semantically related. This score shows high consistency with human judgements in terms of topic quality (Mimno et al., 2010), where higher scores indicate greater topic coherency. As baselines, we used NMF (Lee & Seung, 1999) as the matrix factorization topic model, and LDA+TWE (TWE-1)[5], Topic2Vec[6], with CLM[7] as the benchmark for various combinations of topic model and word embedding model We set the iterations of the Gibbs sampler, parameter update or epochs to 200 for all models except TAN, where the first 50 iterations were used to burn in the Gibbs sampler, where CLM learns word embedding representation using the matrix factorization. We varied the number of top ranked words, and measured their performance by using the coherence model function of gensim[8] with "u_mass".

[2] `https://huggingface.co/datasets/amazon_reviews_multi`
[3] `https://www.yelp.com/dataset/download`
[4] https://pytorch.org/
[4] `https://huggingface.co/transformers/pretrained_models.html`
[5] `https://github.com/largelymfs/topical_word_embeddings`
[6] https://github.com/ukgovdatascience/topic2vec
[7] https://github.com/XunGuangxu/2in1
[8] https://radimrehurek.com/gensim/models/coherencemodel.html

Table 2: Comparison of topic coherence: N is #ranked words. The dimensionality of the embedding space for Topic2Vec, CLM and TAN is set to 100(Amazon)/200(Yelp), the skip length, and #negative sampling is set to 5, and 5, respectively. The values in bold show best performance, where the bold value denotes the statistical significance for $p < 0.01$, compared to the best baseline.

	Amazon			**Yelp**		
N	5	10	20	5	10	20
LDA(TWE)	-2.11	-2.26	-3.33	-2.93	-3.08	-3.76
NMF	-1.92	-2.54	-3.23	-2.67	-2.91	-3.82
Topic2Vec	-1.75	-2.13	-2.61	-2.28	-2.36	-2.71
CLM	-1.71	-1.82	-2.26	-1.52	-1.82	-2.22
TAN	**-0.88**	**-1.23**	**-1.77**	**-0.92**	**-1.45**	**-1.01**

Table 3: Comparison and Ablation analysis of various unconditional text generation models: In this table, F, P, D, B and M denotes Frequency, Dist, BLEU, and METEOR respectively. Due to the space, we show only BLEU-2 and Dist-2. GPT2-f, GPT2 with pt is fine-tuned GPT2, and GPT2 with prefix-tuning. The meaning of bold value is equivalent to Table 2.

Data	**Amazon**					**Yelp**				
Model	F ↑	P ↓	D ↑	B ↑	M ↑	F ↑	P ↓	D ↑	B ↑	M ↑
GPT2-f	3.11	27.32	0.73	10.02	7.82	2.65	32.32	0.68	9.03	7.22
GPT2 with pt	3.08	27.38	0.73	10.02	7.88	2.63	32.34	0.68	9.03	7.25
TAN	**3.43**	**25.14**	0.77	10.11	**8.82**	2.78	**30.01**	0.72	9.14	**7.65**
TAN-TA	3.15	26.88	0.74	10.09	8.12	2.69	32.11	0.69	9.09	7.33
TAN-TEM	3.22	26.19	0.75	10.09	8.31	2.72	31.23	0.69	9.09	7.42
TAN-TDM	3.22	26.18	0.75	10.09	8.35	2.73	31.22	0.69	9.09	7.44

Table 2 shows that the top words of learned topics are semantically coherent, which coincides with the finding that using word embeddings improves the quality of latent topic models (Liu et al., 2015; Xun et al., 2017; Nguyen et al., 2015). Compared other models, TAN shows significantly better results, which implies that learning topics using the pre-trained model and topic assignment helps to discover more coherent topics. More precisely, TAN learns topics using the order of both word and topic in a given corpus in each document and updates them iteratively, groups semantically related words more efficiently than the others, and yields more distinct topics than the others.

5.3 Text Generation

To evaluate the quality of generated texts, we add the latest adaptation approach, GPT2 with prefix-tuning (Li & Liang, 2021), and use the well-known evaluation as follows:

Automated evaluation. We used test-set perplexity, Dist (Li et al., 2016), BLEU-N (Papineni et al., 2002), and METEOR (Lavie & Agarwal, 2007) metrics to measure the performance. As perplexity is an automated measure of fluency, though its effectiveness has been questioned in open-domain text generation (Liu et al., 2016), we use the well-known test-set perplexity using different pre-trained NLMs. Then, we use BLEU to evaluate how many n-grams in the generated text overlap with those in the reference text, and Dist to measure the proportion of unique n-grams in the generated text, where this scores for the distinct 1-2-3-grams (measured across all samples generated for a given attribute control task, e.g. a specific topic for topic control). METEOR aligns the output text to the reference text and calculates sentence-level similarity scores for the alignments.

Human evaluation. We employ fluency testing on attribute relevance as the human annotation (Dathathri et al., 2020). Annotators are asked to evaluate the fluency of each individual sample on a scale of 1-5, with 1 being "not fluent at all," and 5 being ,"very fluent," as done in (Lample et al., 2019).

Table 4: Case study of texts generated from models with Amazon: We gave seed words "I am disappointed in this purchase", and show the generated text from each model.

Ground Truth	I am disappointed in this purchase. I bought one of these in another color and in size XL and it fit great.
GPT2+fine-tuning	[I am disappointed in this purchase.] The color is not as vibrant as I would like. It does however still look great. I will use this again
TAN	[I am disappointed in this purchase.] I ordered an XL size in black which arrived with a large hole. There's no way anyone could get this out

For evaluating unconditional text generation task, we set the maximum length of generated text to T=64 for all models, as the average of each text used in fine-tuning data set is around 60. Note that the ground truth texts were excluded from training/validation data to prevent leak of information. As shown in Table 3, TAN outperformed the baselines and achieved better performance over both data sets. These results can be explained by injecting topics and their alignment, which is newly utilized by TAN. A manual error analysis shows that some instances marked as errors were in fact assessed correctly by allowing partial matching of words in a text. When the ground truth text is personalized, human judgement is difficult even if the generated text is different from the ground truth, shown in Table 4. After checking the errors at the word level, we found that the generated text included more abstract or higher frequency words than the reference sentences.

5.4 Ablation analysis

To investigate the respective contribution of TAN, (i.e., TA, TEM and TDM), we conduct an ablation analysis. We remove different components introduced in TAN and show the text generation quality in Table 3. This table shows that the setting of TAN achieves better performance across both datasets. An within table comparison shows that topic alignment (TA) is effective. By comparing the effects of TEM and TDM for the same pre-training data, these newly introduced task improve text generation performance, and make a similar contribution to TAN. In conclusion, this analysis shows that all elements introduced, especially topics, are effective in text generation.

6 DISCUSSION

Previous NLMs use local context information through word sequences to capture semantic meaning, and our approach extends these models by introducing topics as global information through the topic. Because topics are based on word co-occurrence in each text, this information varies with the dataset. While TAN enhances the quality of topic models and NLMs by incorporating topics, its quality greatly depends on the quality of discovered topics and topic alignment. Just as the topic models have enjoyed success in areas other than text processing, this fine-tuning training of pre-trained NLMs via topics could be applied to other tasks. As shown in Table 1, the reason why the difference between Z and K is small is because pre-trained NLMs do not incorporate the topic, and increasing the difference would not only diverge learning but also lose the information they have.

7 CONCLUSION

This paper proposes TAN to adapt pre-trained NLMs to the unconditional text generation task within the target domain. TAN introduces "topic-alignment (TA)", and new tasks such as Topic Distribution Modeling (TDM) and Topic Embedding Modeling (TEM) to discover topics and feed them to pre-trained NLMs. These components enable TAN to discover the target domain specific topics, and fine-tune pre-trained NLMs by feeding these topics. Experiments show that TAN can tailor previous pre-trained NLMs to generate valid text reflecting a given small fine-tuning data set.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalizationg. *CoRR*, abs/1607.06450, 2016.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, Mar 2003.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, pp. 2978–2988, 2019.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *ICLR*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACLM*, pp. 4171–4186, 2019.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In *NeurIPS*, pp. 13042–13054, 2019.

Alexey Dosovitskiy, Lucas Beyer, and etc Kolesnikov, Alexander and. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Noriaki Kawamae. Topic chronicle forest for topic discovery and tracking. In *WSDM*, pp. 315–323, 2018.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. Multiple-attribute text rewriting. In *ICLR*, 2019.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*, 2020.

Alon Lavie and Abhaya Agarwal. METEOR: an automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation, WMT@ACL 2007*, pp. 228–231, 2007.

Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401(6755):788–791, 1999.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *NAACL*, pp. 110–119, 2016.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL/IJCNLP*, pp. 4582–4597, 2021.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *EMNLP*, pp. 2122–2132, 2016.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Topical word embeddings. In *AAAI*, pp. 2418–2424, 2015.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *NeurIPS*, pp. 14014–14024, 2019.

David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *EMNLP*, pp. 262–272, 2010.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pp. 807–814, 2010.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. Improving topic models with latent feature word representations. *TACL*, 3(4):299–313, 2015.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pp. 311–318, 2002.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword fifth edition, June 2011. LDC2011T07.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, and etc. Attention is all you need. In *NIPS*, pp. 5998–6008, 2017.

Zhengjue Wang, Zhibin Duan, Hao Zhang, Chaojie Wang, Long Tian, Bo Chen, and Mingyuan Zhou. Friendly topic assistant for transformer based abstractive summarization. In *EMNLP*, pp. 485–497, 2020.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, and et. al. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.

Guangxu Xun, Yaliang Li, Jing Gao, and Aidong Zhang. Collaboratively improving topic discovery and word embeddings by coordinating global and local contexts. In *KDD*, pp. 535–543, 2017.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *NIPS*, pp. 5754–5764, 2019.

A Appendix

You may include other additional sections here.