Transgenerators

Arip Asadulaev

Abstract

In this paper we introduce an sequence generative architecture based entirely on feed-forward neural networks. Transgenerator conception is based on adversarial learning and transformer model for sequence to sequence translation. Our model includes two functions, generation and translation. The translation function carried out from sentences that have some errors in the construction, to the same sentences without errors. The translation task is well matched by the generation process, in this case the incorrectly generated sentence can be corrected by the same model. Results are demonstrated on (2016) WMT'14 English incorrect-English correct translation and generation and on IMDB dataset.

1 Introduction

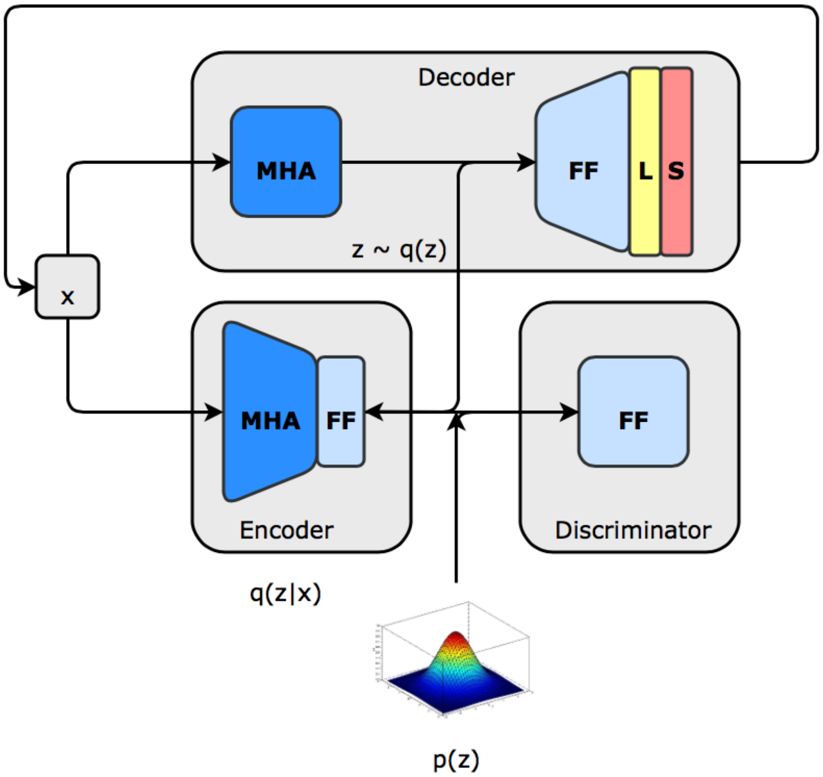

Figure 1: Transgenerator architecture

Preprint. Work in progress.

The most common language translation model is Recurrent Neural Networks[13] (RNNs). Often RNNs is using in machine translation models based on encoder–decoders architectures [6, 8]. In such architecture, an encoder part of model, encodes a source sentence $x = (x_1, ..., x_m)$ into a some vector $z = (z_1...., z_m)$ with fixed size, while decoder is outputs a translated sentence $y = (y_1, ..., y_n)$ given this encoded vector z.

Also RNN are shown impressive results in text classification[10] and generation[4]. RNN generative models are typically trained with maximum likelihood. Models trained with the maximum likelihood are generate sequences by sampling from output distribution. On each step, distribution conditioned on the hidden state and previous symbol generated by model. However, because of gradient vanishing and explosion (Hochreiter et al., 2001) RNNs are difficult to optimize. Also it was shown that LSTM language models use 200 context words on average (Khandelwal et al., 2018), indicating room for further improvement.

Other approach in sequence generation with RNNs is using Generative Adversarial Networks (GANs)[5] framework.

Although GANs were not initially applicable to discrete data due to non-differentiability, approaches such as SeqGAN [9], MaskGAN[12]. SeqGAN successfully combines GANs and RL to apply the GAN framework to sequential data[11]. Where the output of the discriminator is used as reward for the generator.

Professor Forcing is an enother example of combination RNN and GAN. This algorithm make dynamics of RNN hidden states indistinguishable whether the network in free-running generative mode, when its inputs are self-generated and when it trained with teacher forcing mode.

MaskGAN The idea of MaskGAN model is that input have a masked token and target tokens (where the hidden token is replaced with a original token). Output of model passed to the discriminator which may be either real or fake. Adversarial training is applied by REINFORCE(cite) algorithm.

The Transformer[2] and ConvS2S [7] architectures presents not RNN based sequence to sequence translation methods. ConvS2S encodes latent representations in parallel for all input and output positions by using convolution neural networks. The Transformer is using attention for computing hidden representation, where attention[3] compute weighted sum c_i of $(z_1...., z_m)$ at each time step, allowing thus model to focus on different parts of the input sequence.

In this paper, we propose a model which combines the best encoder-decoder based machine translation architecture and GAN framework for better sentence generation. The great advantage of such architectures is that they can perform the translator functions simultaneously with generation function. Now translator functions are used here not for translation from one language to another, they are used to translate within a single domain, from sentences with grammatical error to more correct ones.

2 Background Work

Adversarial Autoencoders GANs takes x and random noise $\epsilon \sim p(\epsilon)$ as input, and are trained to minimize the divergence between a real data distribution x and the distribution z where $z \sim q(z|x)$ is defined by a generative model G.

Alireza Makhzani et al introduce autoncoders with GAN objective in the latent layer [1]. In Autoencoder model, adversarial training is used to match $q(z|x)$ to an chosen prior $p(z)$. The cost function for matching $q(z|x)$ to $p(z)$ is:

$$\mathcal{L}_{prior} = \frac{1}{N} \sum_{i=0}^{N-1} [\log(1 - D(z_i))] \tag{1}$$

where $z_i \sim p(z|x)$, and D is a discriminator. In AAE settings discriminator is training to classify latent samples drawn $p(z)$ from $q(z|x)$. The cost function for discriminator $D(z)$ training is

$$\mathcal{L}_D = -\frac{1}{N} \sum_{i=0}^{N-1} [\log(D(z_i))] - \frac{1}{N} \sum_{f=N}^{2N} [\log(1 - D(z_f))] \tag{2}$$

where $z_f \sim p(z)$. We present a model based on this type of training, but in contrast to the Adversarial Autoencoders (AAE) model, for generating discrete values.

Transformer The Transformer replacing the recurrent layers with multi-headed self-attention layers, with positional encoding. Positional Encoding allows model to make embedding of the input words, with respect to it position in the sentence.

Self-attention mechanisms allows model to look at the sequences using only attention functions. After embedding input sentence, for each word in the embedded sequence, mechanisms of self-attention are applied. Multi-headed attention is a extended version of self-attention mechanism, which consist of several self-attention. Each of self-attentions in Multi-headed attention performs a different function, as a result, the concatenated total output of the layer contains more complete information about the input sequence.

Among other things, Transformer is the first model from encoder-decoder family based entirely on attention. At each time step Transformer model also optimized to maximize the likelihood estimation of the ground word . This model beats RNN based translation architectures, and demonstrated state-of-the-art results on the English-French and English-German translation tasks(cite).

3 Transgenerator

3.1 Method

One of the main problems in text generation is the presence of grammatical errors, and errors in sentences permutations(cite). In this section we present upgraded version of Transformer model(cite) which consist of encoder, decoder, and discriminator part for adversarial learning. To create sentences containing grammatical and syntactic errors, a few words in the sentence were mixed randomly. In view of the fact that not all sentences with mixed words are incorrect, we used a sentence check with a parse grammar tree.

Translation: The first stage is learning to translate from wrong to correct sequences. To solve the problem of translation, for every correct sentence x sampled some modifications in the wrong direction (x'_0, x'_1, x'_2). In principle, any form of encoder-decoder model can be used as a translator, including the RNN based sequence-to-sequence model. This model translation part is a transformer, which improve the quality of translation, using the attention mechanisms. Mechanisms of attention it's very important in bad-to-good translation problem, because it's allow this model to more carefully.

Generation: For adversarial training, a discriminator is added to the translation model. Figure(1). In a model with a feedforward encoder and decoder, the network input can be represented as x, and $p(x)$ the model distribution, while the data distribution can be presented as $p_d(x)$ (??). Having z as a latent vector, encoding distribution is the $q(z|x)$, decoding distribution is the $p(x'|z)$, where x' is target sequences, and $p(z)$ is the prior distribution.

The generator part of transgenerator is the encoder of the transformer $q(z|x)$. Transgenerator attempts to minimize the translation error and guides $q(z)$ to match $p(z)$. An aggregated posterior distribution of $q(z)$ defines by transgenerator as:

$$q(z) = \int_x q(z|x) p_d(x) dx \tag{3}$$

The encoder tries to emit the aggregated posterior distribution $q(z)$ which can be classified by discriminator as true prior distribution $p(z)$.

In the result of the training, encoder learns to convert the data distribution to the prior distribution, while the decoder learns to restore sequences from a given distribution.

Transgeneration: Eventually having a part of the generator, and part of the translator, model can be trained with dual objectives – a traditional error criterion, and an adversarial training criterion[5] that matches the aggregated posterior distribution of the transformer latent representation $q(z)$ to an arbitrary prior distribution $p(z)$. Having such capabilities, the model is able to generate, sequences, and if there are some errors in them, the generated sequence can be fed to the input of the encoder.

3.2 Architecture

Transformer This part of model consist of a multi-head self-attention mechanism, and position-wise fully connected feed-forward network, There are totally 6 layers are in encoder. Each of the two sub-layers connected with residual connections, with layer normalization(cite) all layers in the model, produce outputs of 64 dimension.

In decoder layers inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. In decoder layers, self-attention sub-layer is modified with mask, this is employed to prevent positions from attending to subsequent positions. Finally, as in encoder, decoder architecture consist of 6 layers.

Discriminator Discriminator is a simple feed-forward neural network with input size of encoder embedding size, 2 hidden layers size of 512 neurons with RELU activation functions, and sigmoid on output layer neuron.

4 Experiments

4.1 IMDB dataset

For toy test we choose 40 length sentences from IMDB dataset, with only top 2000 words vocabulary size. Also, model was trained on Full IMDb dataset, which consists of 25,000 labeled, and 50,000 unlabeled training instances. First 40 words of each review was used for model training. The generation training was held in parallel with the model translation training.

Table 1: Perplexity and Accuracy of model on IMDB, and WMT 14 dataset. Where IMDB 0 is dataset with max length 40 and IMDB 1 dataset with max length 10.

Model	Dataset	Perplexity	Accuracy
Transgenerator	IMDB Toy	3.85523	95.385
Transgenerator	IMDB Full	7.59780	83.625

Table 2: Random chosen examples of Conditional Translation, after 25 epoch of training on toy IMDB dataset

- i care the - or any after all any era that can be - down and is - by realism it's a good film even the best one in its its - but soon message indeed unusual unusual joke.
- of - and i think you should it should not read be put back on tv channel this movie will deserve a - - it will the real career i'm playing now a movie on video on dvd dvd
- is extremely boring and disappointing film 'many words as each - make for them in an extreme - manner its not funny moments and ready to to be - and you'll have very much you need this one
- but to add some help of humans - sounds rubbish and that's saying a picture - - yet it's anti women - them like - - things to be - and entertainment it is from beginning to finish it
this movie is to make you laugh but then a few minutes are ready you to think and completely understand personal to help next i think its a good movie that will continue right by its bad taste and

Table 3: Random chosen examples of transgenerated samples, after 25 epoch of training on toy IMDB dataset. Same model was used for 10, 20 ,30,40 length generation

- and actors that created a wonderful film and -
I saw it in jerry star a deeply bored to me and - the award me x - -
- villain and - - random characters and make it direct to the below low budget - - pacing herself is x - - and - rape their - called
- almost real fashion for the cast is below any - - i would share the cast as - - and yes it comes to - toward half - - - uncle - ray - - - - - -

Table 4: Results on IMDB dataset, Mask GAN trained to fiil sentence in the end of the sentence, since our model trained to replace random words in sentence

Model	PPL
MaskMLE	273.1
MaskGAN	108.3
Transgenerator	7.59

Table 5: Random chosen examples of Conditional Translation, after 25 epoch of training on full IMDB dataset

Movie is funny in places it needs to be and be relatively well reminiscent of christopher christopher walken if you liked those writer adaptation then i recommend you keep you look out and this one br br performances performances
Making imdb for reviews before but i think that will not anymore this is ridiculous i have been done in town quite far but many times not enough lines you can cancel my account your site of a pain pain
Evil guy this movie would be a 0 out of 10 the bird and makes it a 3 if you are truly bored or want something terrible to really rent this and forward high to the final battle battle
A lot of other better choices joseph blind is really kinda great because it gives you thrills and chills on a upcoming star power but does it in a way that is completely fresh and definitely totally running me
One note as funny but there is no bruce lee so enter the ninja is an even worse film until now this is the second the big comedy party party and the beautiful film i gave 1 10 to 10

Table 6: Random chosen examples of transgenerated samples, after 25 epoch of training on Full IMDB dataset. Same model was used for 10, 20 ,30,40 length generation

I best marvelous singer unit show since was late j
Better this than storyline com if world play action did how through picture did tale since i business this with
Better flashbacks g on best best horror are factory was a least was police giving also best best understood his took to his his critics touches married beliefs flashbacks
Well although change the you've other him there's performance provide in the lead better and see forgotten that's where better and sets come their everything do probably amazing how you forgotten chris before guess up the back probably the him

5 Conclusion and Future Work

In this article was presented the architecture allowing to generate sequences using feed-forward networks. In addition to the generative ability, the network has the ability to correct incorrect sequences. The results can be interpreted in different ways. Despite this, the main idea in this article is that the model that generated something should have the ability to make corrections.

In the view that the Transformer model allows processing characters of a sequence in parallel, this allows us to form a memory for storing sentences embedding and not tokens.

References

[1] Makhzani Alireza, Shlens Jonathon, Jaitly Navdeep, Goodfellow Ian, and Frey Brendan. Adversarial autoencoders. *Arxiv*, 2015.

[2] Vaswani Ashish, Shazeer Noam, Parmar Niki, Uszkoreit Jakob, Jones Llion, Gomez Aidan N, Kaiser Lukasz, and Polosukhin Illia. Attention is all you need. *Arxiv*, 2017.

[3] Bahdanau Dzmitry, Cho Kyunghyun, and Bengio Yoshua. Neural machine translation by jointly learning to align and translate. *Arxiv*, 2016.

[4] Alex Graves. Generating sequences with recurrent neural networks. *Arxiv*, 2013.

[5] Goodfellow Ian J, Pouget-Abadie Jean, Mirza Mehdi, Xu Bing, Warde-Farley David, Ozair Sherjil, Courville Aaron, and Bengio Yoshua. Generative adversarial networks. *Arxiv*, 2014.

[6] Sutskever Ilya, Vinyals Oriol, and Quoc V. Le. Sequence to sequence learning with neural networks. *Arxiv*, 2014.

[7] Gehring Jonas, Auli Michael, Grangier David, Yarats Denis, and Dauphin Yann N. Convolutional sequence to sequence learning. *Arxiv*, 2017.

[8] Cho Kyunghyun, Merrienboer Bart van, Gulcehre Caglar, Bahdanau Dzmitry, Bougares Fethi, Schwenk Holger, and Bengio Yoshua. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Arxiv*, 2014.

[9] Yu Lantao, Zhang Weinan, Wang Jun, and Yu Yong. Seqgan: Sequence generative adversarial nets with policy gradient. *Arxiv*, 2016.

[10] Miyato Takeru, Dai Andrew M, and Goodfellow Ian. Adversarial training methods for semi-supervised text classification. *Arxiv*, 2017.

[11] Salimans Tim, Goodfellow Ian J., Zaremba Wojciech, Cheung Vicki, Radford Alec, and Xi Chen. Improved techniques for training gans. `http://arxiv.org/abs/1606.03498`, 2016.

[12] Fedus William, Goodfellow Ian, and Dai Andrew M. Maskgan: Better text generation via filling in the. *Arxiv*, 2018.

[13] Wu Yuhuai, Zhang Saizheng, Zhang Ying, Bengio Yoshua, and Salakhutdinov Ruslan. On multiplicative integration with recurrent neural networks. *Arxiv*, 2016.