Context Aware GAN for sequence text generation in tensor-flow lite for android AI chat application

Prashant kaushik
Prashant.kaushik@jiit.ac.in

Sunil Prasad
sunil.bhilai@gmail.com

1.Abstract

A Novel model and technique to train the generative adversarial networks (GAN) that uses a generator based model by feedback operation through discriminator with contexture-1 and contexture-2 to give impressive success in the context of text generation for android tensor-flow lite applications. In this paper we are adding two contextures for generating the more context oriented results to get desirable text while measuring for the confidence score. Previous GAN models are basically discriminator oriented model in which training of generator for token generation is going on untill or unless we get desirable result given by discriminator. But in our GAN model we are taking sentences from corpus and assigning them a particular numbering using pre-process step method for giving percentage wise corrected result by generator and the process continues till we get appropriate context or a combination of cascaded context which is cascade of the context vectors of both the contextures. However using of contexure is continuously increasing the confidence of sentence in the field of long text generation. We have compared our results of training and model innovation. The correctness of the results of the context for the android app of the lite tensor-flow version is validated by over 20 humans as the android application containing the trained model was distributed for testing in the campus where people can chat to the model using their android app and can submit the context orientation of the results from their android device only. We have also opened the source code and the trained model of the tensor-flow lite app on github for further participation of research community.

2. Introduction

Generative model of text generation is an essential tool for natural language processing (NLP) because it contains the foundation for many applications like dialogue generation, machine translation [16], text summarization [16], table summarization [16]. In the text generation model is totally based on sequence-to-sequence [17] pattern is known as end-to-end model in which sentence code is encoded and target code is decoded by using end to end model. It takes true data from real world and generated data by generator which is train to discriminator convolutional neural network (CNN) to display the output. Basically, policy gradient used in seqGAN uses a Monte Carlo (MC) search along with a roll-out policy to sample data to the remaining input and uses the state-action value in intermediate step before passing to MC search. However, seqGAN [10] have not much effective performance for a long text generation. Because seqGAN decision-making model and recognized the generative model by using gradient based Reinforcement learning policy.

Variational Autoencoder (VAE) train the hierarchical latent variable generative model. In which two types of processes are present we can name them training and generation process. In training process encoder takes the input in the context of inference encoded vector in latent space and in generation process takes sampled vector from the latent space then this latent space passes through latent distribution (hidden layer) to the decoder to reconstructed input or generated content. When we discussed about the text data latent variable is considered as a sentence representation to guide the generation process. VAE framework is mainly consist of convolutional layer as a encoder and deconvolutional layer as a decoder to show the representations of generated sentence by combining with recurrent layer via using feed forward path model architecture. In this method we focus on the sentence representation via using sentence encoders could be generate the effective model for text generation that is useful for natural language processing (NLP) tasks. In a manner we make a set of sentences getting from large sentence corpus data for training purpose. We model this using distribution using Generative Adversarial networks (GAN) [13],[17], to make generative model of sentence for giving promising result with using of conditional Gated recurrent unit

(GRU) [17] is advanced version of long text generation (LSTM) based language model. It is encoding the high-level sentence to provide better information to decoder.

Currently various neural network models are working to the task of question answering (Q/A) and reading comprehension [3],[10]. This model is mainly depending on the labeled data. However, it is very difficult to store question answer data set in a large amount. Normally most of the question answering dataset have only thousands of questions answering pairs such as Web questions [18], MCTest [19], WikiQA [14], and TREC-QA [1], Although a larger question answering data set with more than the thousands of questions answering pairs such as SQuAD [20], MSMARCO [21], and NewsQA [22], Storage of large data collection is a heavy task. This task hampers the real-world application for domain specific question answering.

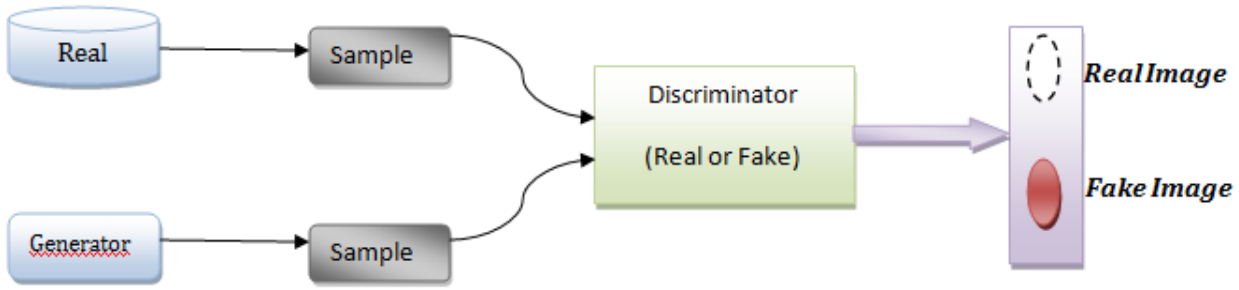


Figure 1: - A typical GAN Model For text generation

To better support of unlabeled data of novel structure, GDANS proposed a new model is called Generative Domain-Adaptive networks (GDANS) [23]is used semi-supervised question answer data set. The starting point of our framework is extracting the split form of possible answer in the context of unlabeled data and then train the generator model to split the unlabeled data in terms of context. The model generated model and human generated question answer pair is called as discriminative model which is conclude the data for long text generation. There is different model technique to model generated data distribution and human generated data distribution discrepancy we used to two different types of domain techniques to train generated data distribution first name type is *domain tag* to check the question answer model is model generated or it is human generated to learn the discriminative model that model generated text is correct or incorrect or in other words say it is domain related text. And the second one is reinforcement learning to guide the generative model o control the adversarial function of discriminative model. In condition on that way we analysis unlabeled data. In that kind of experiment GDANs is very effective structure for lead to labeled data when it is available.The focused works are from GDANs architecture, a) they proposed a new data structure to produce via using different kind of model to generate the long text generation , b) their model to extend the conditional generation model for text generation. In other way we can say that to train a conditional GAN, c) model is trained the framework of question answer model to train the framework based on discriminator and generator architecture to acquire the high-density data policy on the basis of using of hierarchical data for long text generation.

3. Our Approach

Our approach is divided in various section mentioned below with their details like the main implementation is done in tensor-flow 1.4 (as it supports tensor-flow-lite conversion fully). Then conversion process of trained model which is very important as we have to keep app the input and output dimensions intact. Then use this model into android application which input a noise by the user and generate two contexture output and submits the user reaction on to our server which is running on firebase a server frame provided by Google.

3.a TCGAN (Two Contexture GAN)

Now our model is slightly changed with respect to previous GAN model that our model basically based on text generation in which we take input as a text in terms of sentences taken by various sources like book, comments, subtitle of movies etc. to make a corpus and we take unique sentences to form vocabulary with proper numbering and

sentences are replace by the numbers using pre-process step. On other hand we rectify the previous GAN model for impressive result with added two contexture i.e contexture 1 and contexture 2 between generator and discriminator. In which text under go through this contexture to give usable text for discriminator and it gives the result of context from 0 to 1 in a percentage manner correct text not in true or false depends on the flow of process. Then after producing 3 outputs from the model the output with more confidence is send as output. The training process and data preparation process is very intensive and took over 3-4 months for the same.

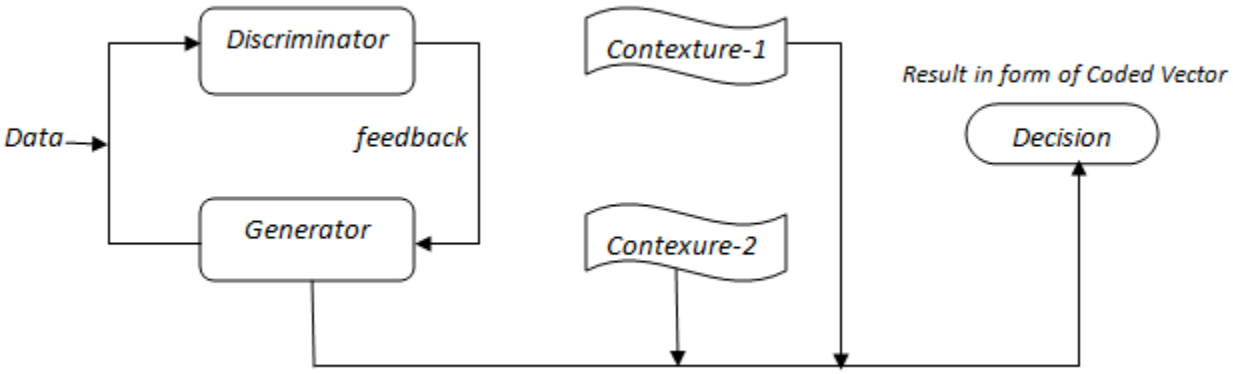
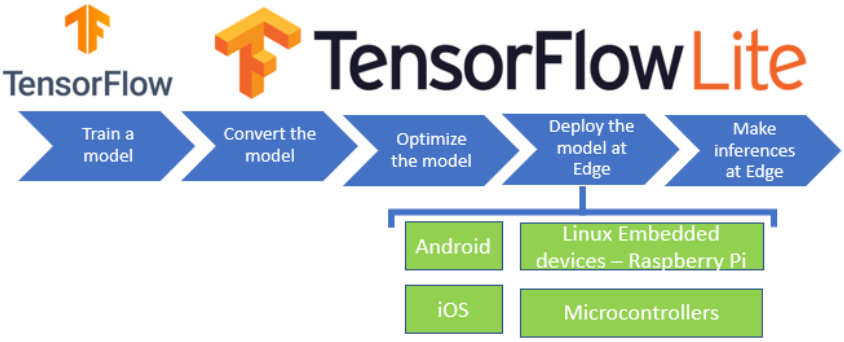


Figure 2: TCGAN(our mode)

Recent success of text generation via using model we represent here which is implies effective result for long text generation with constantly increasing the confidence of the user. In this model we are using contextures which work is constantly improve the percentage growth of the input coming from the generator until the correctness level is desirable with effective training of generator. This training will run until or unless when the result in percentage manner correct text it means result is close to 1.

3.b Tensor-flow to Tensor-flow-lite

Once we are satisfied with training and the results of the model than we make a lite version for it and making all the dimensions same along with the vocab of the corpus and labels. The conversion process is very much required as this chat bots has application in may sectors like Q/A for banks website or apps and other similar use cases. So we need general public views and their reaction to increase the system's acceptability in the market. Plus the tensor-flow-lite only supports limited operations of the base tensor-flow and does not supports ops like RNN and etc so we are limited to CNN only mainly. Here we have to write dedicated python program which checks the input and out dimensions of the main model and then use the tensor-flow methods for lightening the model and then recheck all layers as sometimes this process also changes the internal layers and outputs are not good. So this model is for a python program which does these 3 above mentioned things.



3.c Using Tensor-flow-lite model in Android application.

This process is coding the android app which can use the model and present the user with outputs and also take user reactions and submit them to server which help us in further data analysis and training the model. Below are the some snapshots of the application and link of the code which we have opened sourced for research community for comparison and further enhancements also we plan to make this code so modular so that people can use this as a frame work for testing their text model but that is a future work.

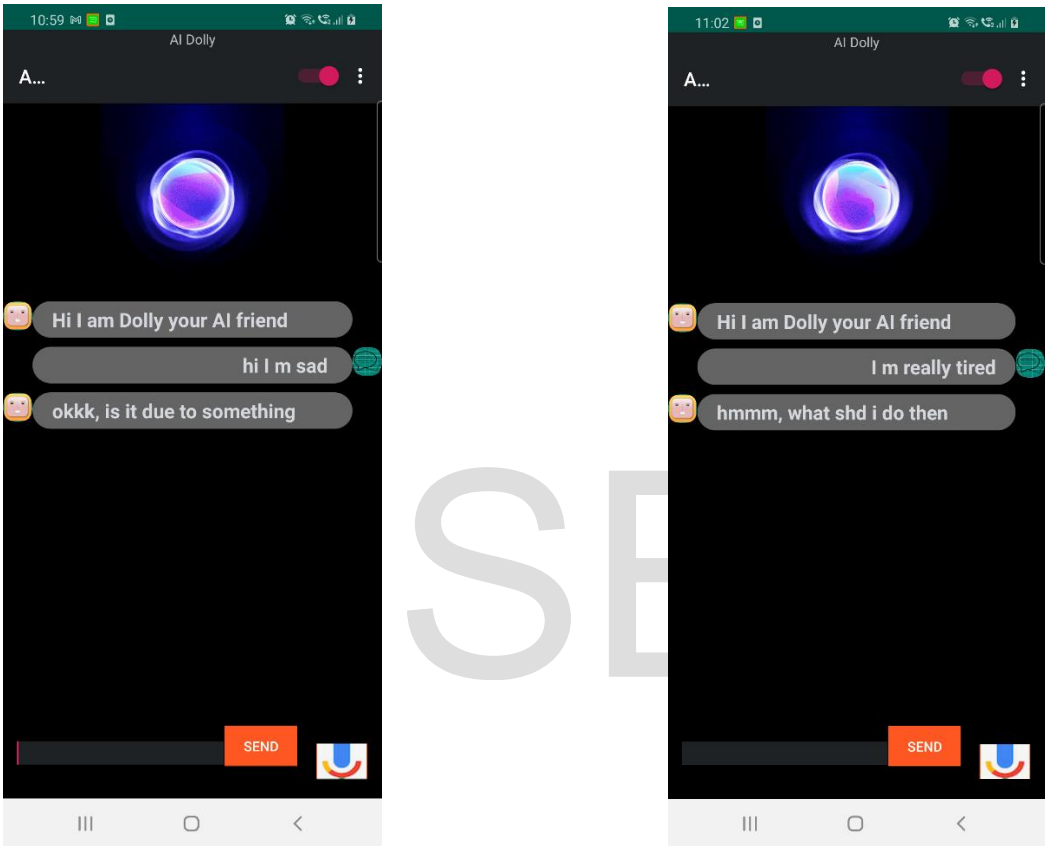


Figure 3: AI-Dolly our android app using the trained lite model.

The android app above sometimes crashes and the cause is the re allocation of memory on every call to the model which we have planned for rectification in future works.
The link of the open code of the android application is here https://github.com/prashant9898k/Ai-dolly

3.d Details about the Contextures

These are two different model where the layers are organized in similar manner but the number of neurons and length of input and out vectors are different along with the different size of batches. The contextures are of 3 layers based where first layer is of about Inverse Conv1D than two layers of Con1D so that we can converge the output vector and find the categorization etc.
The nearness and farness of the context is also calculated and based on vectors outputted by the models and also using eigen vector distance measurement methods these values are used as feedback for training and comparison.

3.e Results of the training from Tensor board

After the training of the model and the contextures we have plotted the sample 10000 output at various training stages to see the clustering of the results and each of these cluster is a different context. We have used tensorboard for plotting these clusters which is inbuild tool in tensor flow to show the results and training checkpoints.

Figure 4: The 10000 output and their clusters at 1k iteration

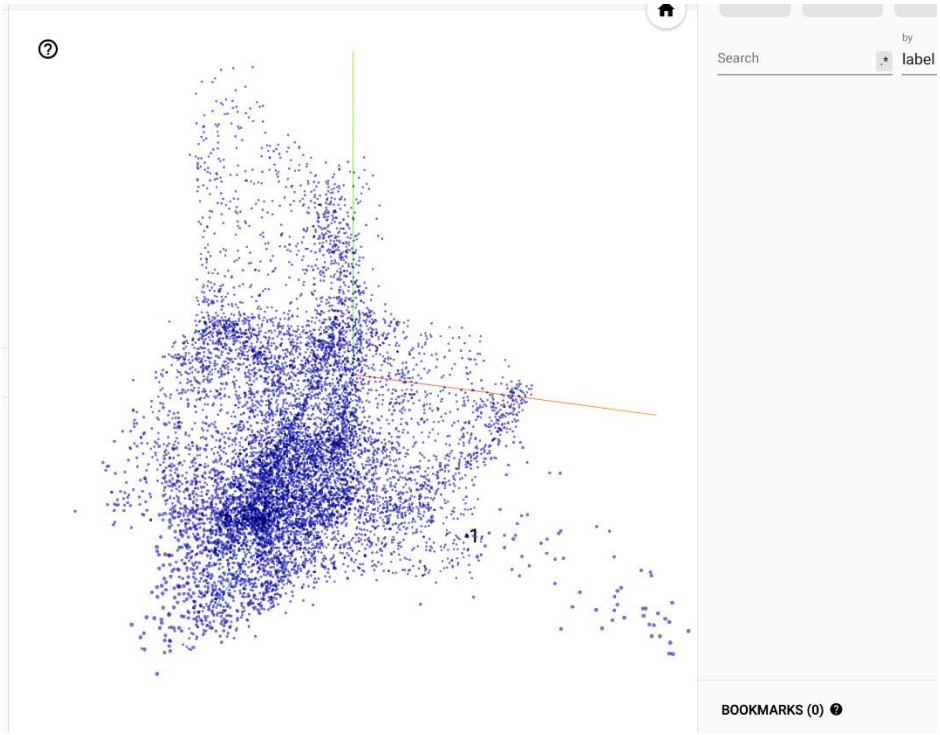


Figure 5: The 10000 output and their clusters at 5k iteration

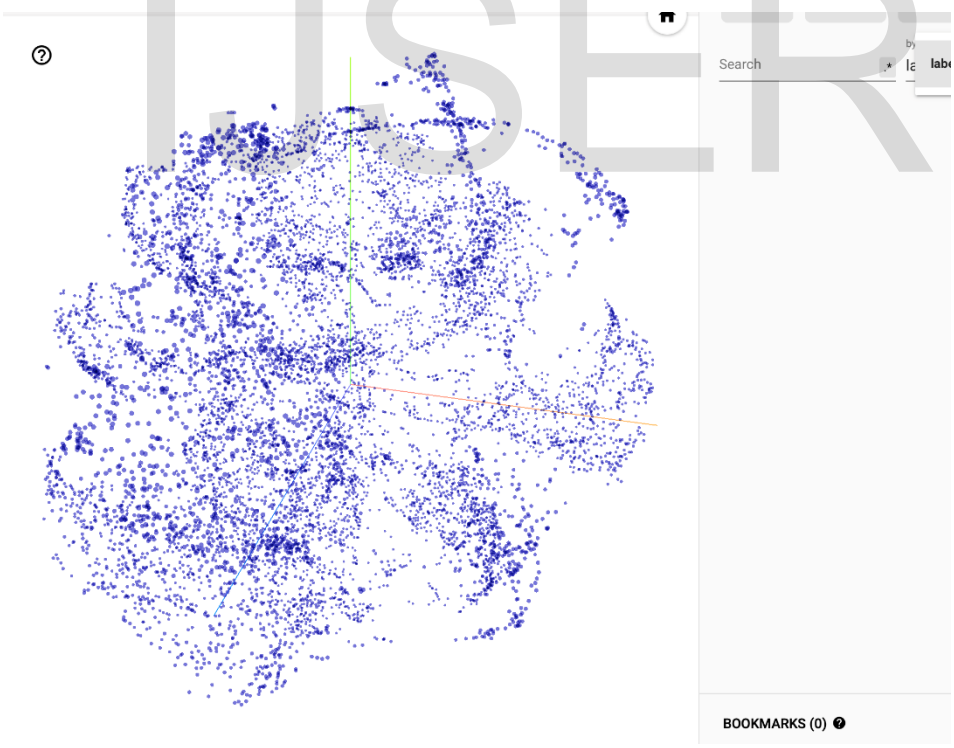


Figure 6: The 10000 output and their clusters at 10k iteration

4. Comparison

We are able to compares some of our model features with the existing model and some old models as we know GAN comes in many shapes and varieties so full comparison is not possible as all the designs have very different parameters and not all of then carry equal weights and comparison if complex in itself.

	Training volume	Feedback method	Computation factor
SeqGAN	10000 words	Reward based	Moderate
TC-GAN(our model)	Over 10000 words	Context nearness	good
Texy GAN	10000 Unique words	Gradient Policy	good

Table 1: comparison with SeqGAN

	Layers	Model Design	Training Cycle
Conv seq2seq	Embedded	Encoder and decoder	Time reversed
TC-GAN(our model)	Embedded	Approximatly Contexture Based	Contextually Based

Table 2: Architectural Comparison

	Public Data set	Methodolgy	Input Data
Conv seq2seq	WMT	CNN Architecture	Pair of conversation
TC-GAN(our model)	Real-world Data set	Contexture Based	Sentences
TexyGAN	MNIST	Hierarchy design	Sentences
SeqGAN	Set	Teacher forcing	One hot vector
Autoencoder	MNIST,CelebA	Multi-Task learning	Latent Vector
IRGAN	Real-world Data set	Minimax game	Textual document

Table 3: Context Comparison on the outputs of models

5. Conclusion & Future Work

In the above work we introduce a new model which use the different type architecture model to give better result than previous GAN model with using two contexture generator and discriminator and we are able to port it to an android application which is running on edge devices. The use of two contexture is better as human usually speaks in two or multiple sentences as a reply for the given context and our model is also doing same as shown in the android app snapshots in Fig 3 above. The model trained above has some pitfalls as shown in the comparison table and also the android app we made has some issue on continuous usage as it keeps loading the model from memory again and again and thereby crashes the app, we have planned to work upon both these issues and also make the app as modular as possible so that anyone in the research community can just plug his/her model and gets the boiler plate code ready along with the memory issue mentioned.

6. References

[1] Dethlefs N and cuayhuitl.H, Hierarchical *reinforcement learning for adaptive text generation. Proceeding of the 6th international conference*, 2010.

[2] Mauldin M.L. *Semantic rule based text generation. Proceeding of the computational linguistias and 22nd annual meeting on association for computational linguistics*, 1984.

[3] Kaiming He, Xiangyu, Shaoing Ren and Jian sun. *Deep residual learning for image recognition, Proceeding of IEEE conference on computer vision and pattern recognition*, 2016.

[4] Yoshua Bengio, Jerome louradour, Ronan cellobert and Jason Weston, *Curriculam learning, Proceedings of the 26th annual international conference on machine learning*, 2009.

[5] Niladri Chatterjee, Anish Johnson and Madhav Krishna, *Some improvement over the blue metric for measuring translation quality for hindi in computing theory and application , ICCTA-07 International conference on IEEE,* 2007.

[6] Lei Yu, Xiang Long, Chao Tong. *Advarsarial Control Automation and Artificial Intelligence. Proceeding of the International conference, 2018*.

[7] Wooho Lee, BongNam Noh, Yeosukim ki moon Jeong, *Internet and distributed computing system, Proceeding International conference 2019.*

[8] Zhiwa Huang, Danda Pani Paudel, Guanju Li, Jiqing Wu, Radu Timofte, Luc Van Gool, *Image and video Processing, Proceeding on International Conference, 2019.*

[9] Dejia Xu, Vihao chu, Qingyan sun, *Computer vision and pattern Recognition, Proceeding on IEEE conference, 2020.*

[10] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu, *SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient*.. In AAAI, 2017.

[11] Tong Che, Yanran Li, Ruixiang Zhang, R Devon Hjelm, Wenjie Li, Yangqiu Song,and Yoshua Bengio. 2017. *Maximum-Likelihood Augmented Discrete GenerativeAdversarial Networks*. arXiv preprint arXiv:1702.07983 (2017).

[12] William Fedus, Ian Goodfellow, and Andrew M Dai. *Maskgan: Better text generation via filling in the* . arXiv preprint arXiv:1801.07736, 2018.

[13] Anirudh Goyal, Nan Rosemary Ke, Alex Lamb, R Devon Hjelm, Chris Pal, Joelle Pineau,and Yoshua Bengio. *Actual: Actor-critic under adversarial learning*. arXiv preprint arXiv:1711.04755, 2017.

[14] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: *A challenge dataset for open-domain questionanswering*. In EMNLP. Citeseer,2015.

[15] Jun Wang, Lantao Yu, Weinan Zhang*, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, Dell Zhang. IRGAN: *A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models*. arXiv:1705.10513v2 [cs.IR] , 2018

[16] Xu, F., G. Harten, D.J. Diner, A.B. Davis, F.C. Seidel, B. Rheingans, M. Tosca, M.D. Alexandrov, B. Cairns, R.A. Ferrare, S.P. Burton, M.A. Fenn, C.A. Hostetler, R. Wood, and J. Redemann, : *Coupled retrieval of liquid water cloud and above-cloud aerosol properties using the Airborne Multiangle SpectroPolarimetric Imager (AirMSPI).* 2018.

[17] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengio *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling* arXiv:1412.3555, 2014.

[18] Jonathan Berant, Andrew Chou, Roy Frostig, Percy Liang, *Semantic Parsing on Freebase from Question-Answer Pairs*, EMNLP, 2013.

[19] Mark L. A. Richardson, Emily M. Levesque, *AS AN IONIZATION PARAMETER DIAGNOSTIC IN STAR-FORMING GALAXIES*, the American Astronomical Society, 2013.

[20] Pranav Rajpurkar and Jian Zhang and Konstantin Lopyrev and Percy Liang, *SQuAD: 100,000+ Questions for Machine Comprehension of Tex*t, arXiv: 1606.05250v3**,** 2016.

[21] Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, Jason Yosinski, Plug & Play Generative Networks: *Conditional Iterative Generation of Images in Latent Space*, arXiv:1612.00005v2, 2016

[22] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, Kaheer Suleman, NewsQA: *A Machine Comprehension Dataset*, arXiv:1611.09830v3, 2016.

[23] Yaroslav Ganin, Victor lempitsky, *Unsupervised Domain Adaption by Backpropagation* arXiv:1409.7495v2, 2014.