Retrieval-based ChatBot using Neural Network

Team

Robiul Hasan Nowshad (Leader)

ID: 160104061

Mail: nowshad21aud@gmail.com

Samia Zahan

ID: 160104057

Mail: samiazahan3@gmail.com

Muna Saha

ID: 160104060

Mail: munasaha369@gmail.com

1.Introduction

A chatbot is a computer program that simulates human conversation through voice commands or text chats or both. In our project we have worked on text chat. The name “Chatbot” which is a short form of chatterbot, is an Artificial Intelligence (AI) feature that can be embedded and used through any major messaging applications.for example ”Swelly” which is probably the most famous Facebook Messenger chatbot. It has millions of users which basically surveys on versatile topics by taking votes from its users. Again, The “eBay” chatbot is the most advanced e-commerce chatbot currently. And is also the most used! It is built for the Google Assistant. Which assists people for the best price of any product around the world.

With The progression of advanced technology, trades and businesses are rapidly moving from traditional to digital platforms to transact with consumers. Convenience through technology is being carried out by businesses by implementing Artificial Intelligence (AI) techniques on their digital platforms. For any industry type,It is important to interact with clients. Customer support for similar queries can be handled in an automated way simply by a chatterbot. It also saves human resources for qualitative tasks. Hence improvement of business branding is possible with minimum effort. All these possibilities of benefits are the motivation to develop such AI tools called chatter bot. Nowadays it is vastly being used in lots of industries to ensure better communication. As it fulfills desirability, feasibility and viability criteria, we are motivated to develop such a tool. Though chatbot is easy to use and costs less time and money to develop but to ensure the goal , the biggest challenge is to understand the user intent and to decode the intent hidden inside the queries and messages. More intelligent Chatbot is more capable of interpreting customer conversation and unveiling the intents.

2. Related Works

Chatbot with data driven approaches has drawn significant attention nowadays.. Existing work along this line includes retrieval-based methods[2] , Retrieval-based models use a repository of predefined responses and some kind of heuristic to pick an appropriate response based on the input and context. The heuristic could be as simple as a rule-based expression match, or as complex as an ensemble of Machine Learning classifiers. These systems don't generate any new text, they just pick a response from a fixed set. Another research followed generation-based methods[3].Generative models (harder) don't rely on predefined responses. They generate new responses from scratch. Generative models are typically based on Machine Translation techniques, but instead of translating from one language to another, we "translate" from an input to an output (response).

Our work is a retrieval based method, in which we study context-based response selection.

In a study of retrieval-based chatbots they focus on response selection for single-turn conversation[4].A bot conversation can be as simple as asking a specific question and getting an answer, or it could be a dialog with a series of interactions.The most basic bot conversation is a single-turn conversation. Where the user asks a question, and bot responds verbally. Recently, researchers have begun to pay attention to multi-turn conversation[5]. Where users can have a multi-turn conversation, in which there's a dialog between bot and the user. Bot responds appropriately to the current question by remembering the intent (or context) established by the previous interaction. In a research [6] they discussed types of bot conversation and also gives an intuition that single turn conversation is much more desirable and feasible for textual chatbot, while multi turn suits for

verbal chatbots. And so we have executed a single turn conversational chatbot following retrieval based method.

3.Project Objectives

Every bot should be built to solve a well-defined use case. The first step to creating a well-defined use case involves gathering market requirements and assessing internal needs to create a well-defined use case. Here we designed our project for the E-commerce domain.

The fundamental task of our system is to interact with clients(users). The clients ask questions or queries and our chatbot should be able to answer them in a relevant way. We can break down our tasks as below.

- Take input from the user: Users must interact with the bot in English language. Otherwise the chatbot will respond , “I do not understand” for not being able to understand.
- Process the input data: For any queries from the user the input data will be processed, to be machine understandable.
    I. Pre-processing:involves tokenization, stemming , filtering.
    II. Featuring data for representation: Converting documents to vectors that can be fed to the model.
- Understanding queries: for the purpose the input data is analysed by the neural network model developed earlier. Tasks under this objectives are:
    I. Train ChatBot: After fixing the domain the next task is to train the ChatBot. Doing so allowed our ChatBot to better understand user utterances, and allowed the engine to better prioritize one ChatBot task or intent over another based on the user input. We train the ChatBot using Deep Learning to improve utterance recognition.

II. Test ChatBot: Once we've built and trained our ChatBot, it needs to be tested carefully and analyse DL and NLP models. It must be ensured that the model has not inadvertently trained using a large number of conflicting utterances while paying close attention to false positives and false negatives. Talk to ChatBot option, Utterance Testing and Batch Testing aid in testing and improving the performance of the Bot.

- Decision making: After analysing the query the model will return a probability of prediction, depending on that the decision (which tag does the pattern/query hold?) is taken. Result is picked if the probability is equal or more than 75%.
- Giving response: Once the tag is selected the bot responses corresponding to the pattern/query of that tag.

Some dummy input output:

```
Let's chat! (type 'quit' to exit)
You: hi
Bot: How dare you address me like that
You: what is your name?
Bot: I dont have a name yet but I was thinking maybe SkyNet. That has a nice ring to it dont you think?
You: what you do?
Bot: I do not understand...
You: tell me joke
Bot: A bear walks into a restaurant and say's I want a grillllllled.....cheese. The waiter says Whats with the pause?
The bear replies Whaddya mean, I'M A BEAR.
You: bad joke
Bot: I didnt expect you to understand my genius comedy. You need a minimum IQ of 200 to even understand the depth of my humor
You: i hate you
Bot: Well thats not very nice
You: quit
```

4.Methodology

The figure below is the workflow of our model:

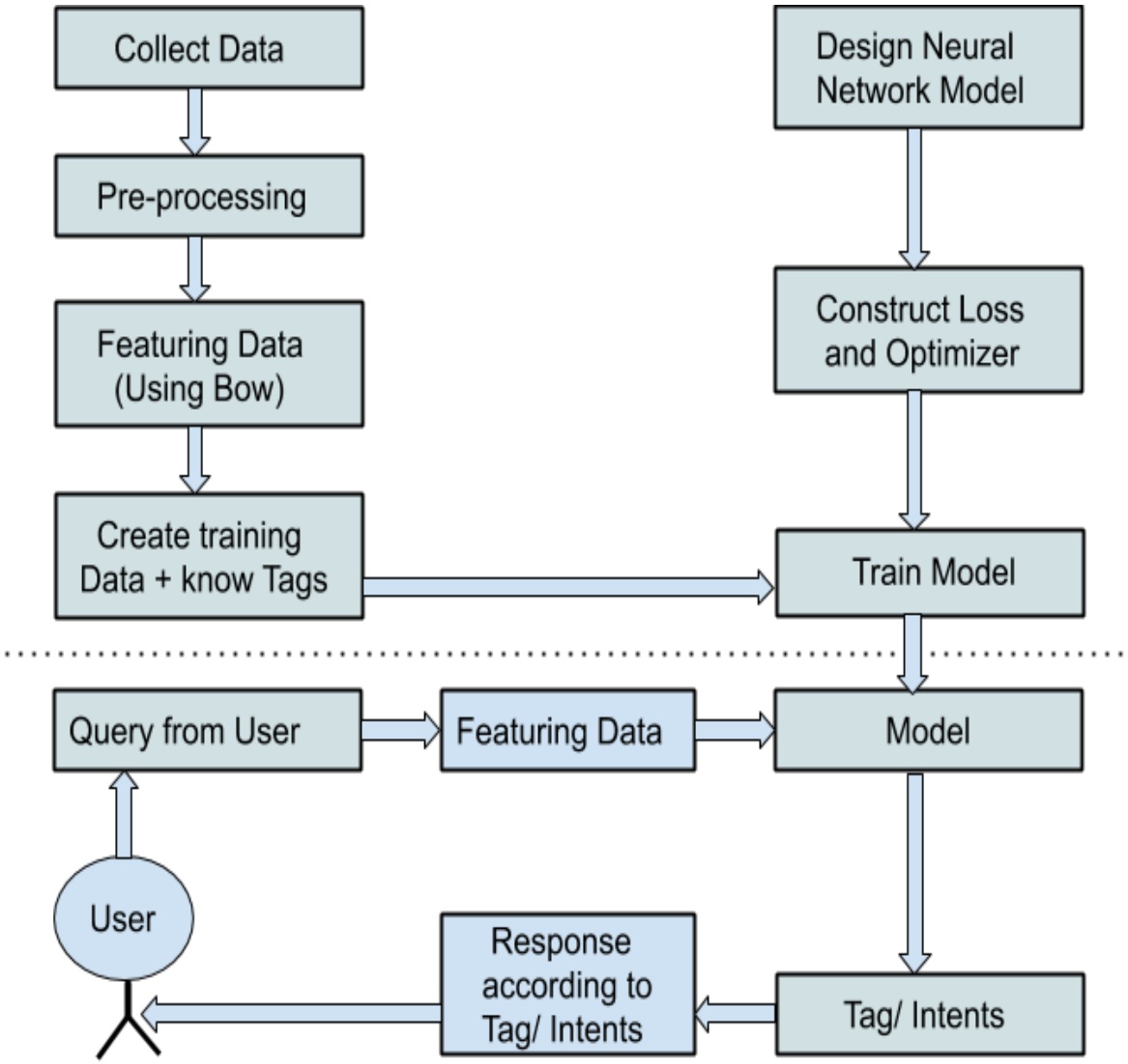

Figure (I): Model of our workflow

Description of steps of Methodology:

Data collect

The dataset we have used in our system is collected from [E1]

Pre-processing

Pre-processing is the primary step required for all Natural Language Processing. It is the process of preparing the data in a readable format. It cleans the data and reduces the noise. We applied basic pre-processing like tokenization, filtering, stemming and stop-word removal.

I. **Tokenization:** Tokenization is a very common task in NLP, it is basically a task of chopping a character into pieces, called a token. On our dataset we have used the punkt tokenizer of python NLTK library, which tokenized the input sentences from user to words.

II. **Stemming**: Stemming is the process of reducing a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma. Basically, the vital goal of the stemming process is to change the words to its root words. We have used PorterStemmer from NLTK library for this purpose.

III. **Filtering:** Extra spaces, punctuation like (‘?’, ‘,’ ,’!’) are filtered as they have less significance on the data.

Featuring data (using BoW)

Data featuring is the process of transforming the data into machine understandable as in form of vectors or numbers. There are a lot of processes for transforming the text data into machine understandable form. In our project we have used the Bag of Words model. In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words.It describes the

occurrence of words within a document. It involves two things: i) A vocabulary of known words(unique words in the whole document). ii)A measure of the presence of known words. And finally returns a vector for each document.

Designing Neural Network Model

Artificial neural networks are built like the human brain, with neuron nodes interconnected like a web. The human brain has hundreds of billions of cells called neurons. Each neuron is made up of a cell body that is responsible for processing information by carrying information towards (inputs) and away (outputs) from the brain. In artificial neural networks, there are 3 kinds of layers e.g. input, output and in between hidden layers.

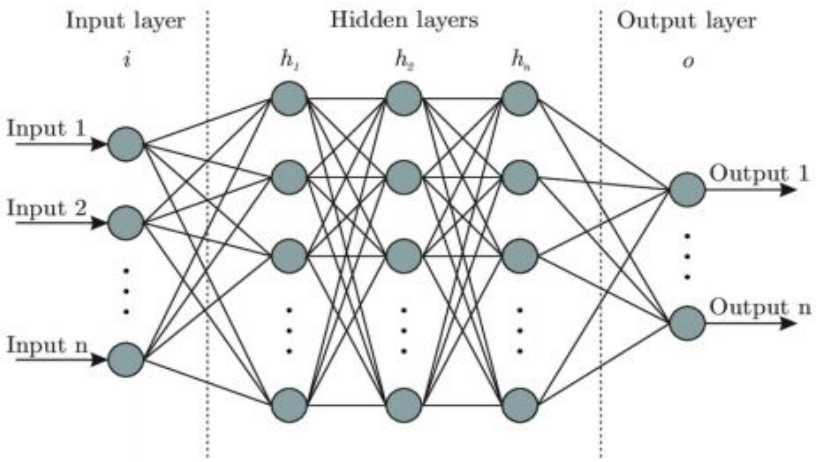

Fig: General Neural Network

Every layer has multiple neuron nodes. Each node has 2 tasks e.g. calculate linear value with weights and bias by a linear function and pass the linear value in an activation function return discrete value as output.

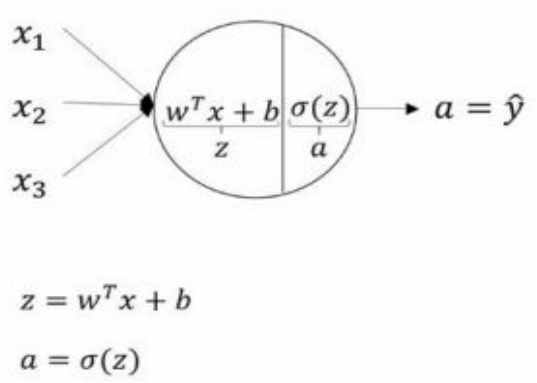

Fig: Single Neuron

We constructed our model with input layer, output layer and two hidden layers. Input layer size was the length of input feature and output layer size is number of classes. Each hidden layer was the size of 8. We used the ReLU activation function.

$$ReLU, \quad y = max(0, x)$$

A graphical representation is given Below,

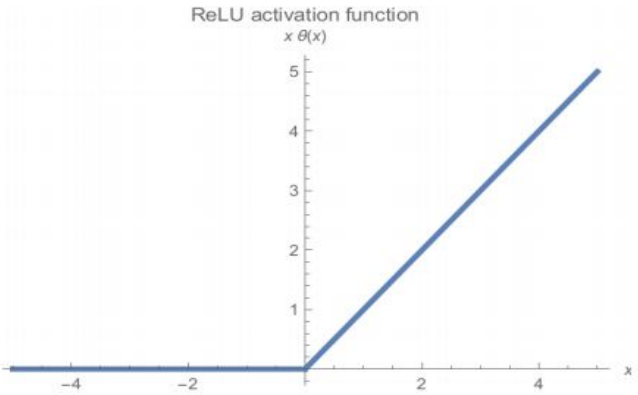

Fig: ReLU function

Loss and Optimization: To train a neural network model, There are two major tasks, one is forward propagation another is backward propagation. In forward propagation we predict the classes and compute the loss/Error. On the other hand, in backward propagation we update weights and bias to minimize the loss. For loss calculation we used categorical cross entropy loss function and for optimization we use Adam optimization algorithm.

Cross Entropy Loss: The categorical cross entropy loss function calculates the loss of an example by computing the following sum,

$$Cross\ Entropy = -\sum_{i=0}^{Output\ Size} Y_i * log(Y'_i)$$

Where, ***y'*** is the ***i-th*** scalar value in the model output, **y** is the corresponding target value, and output size is the number of scalar values in the model output. This loss is a very good measure of how distinguishable two discrete probability distributions are from each other. In this context, ***y*** is the probability that event ***i,*** occurs and the sum of all ***y*** is 1, meaning that exactly one event may occur. The minus sign ensures that the loss gets smaller when the distributions get closer to each other.

Adam Optimization Algorithm: Adam [7] is an adaptive learning rate optimization algorithm that's been designed specifically for training deep neural networks. Adam can be looked at as a combination of RMSprop and Stochastic Gradient Descent with momentum. It uses the squared gradients to scale the learning rate like RMSprop and it takes advantage of momentum by using moving average of the gradient instead of gradient itself like SGD with momentum. The pseudocode is given below.

Algorithm:

$V_{dW} = 0,\ S_{dW} = 0,\ V_{db} = 0,\ S_{db} = 0,$

On iteration T :

Compute dW, db using current mini-batch

$$V_{dW} = \beta_1 V_{dW} + (1-\beta_1)\, dw$$

$$V_{db} = \beta_1 V_{db} + (1-\beta_1)\, db$$

$$S_{dW} = \beta_2 S_{dW} + (1-\beta_2)\, dw$$

$$S_{db} = \beta_2 S_{db} + (1-\beta_2)\, db$$

$$V_{dW}^{corrected} = \frac{V_{dW}}{1-\beta_1^T}$$

$$V_{db}^{corrected} = \frac{V_{db}}{1-\beta_1^T}$$

$$S_{dW}^{corrected} = \frac{S_{dW}}{1-\beta_2^T}$$

$$S_{db}^{corrected} = \frac{S_{db}}{1-\beta_2^T}$$

$$W: = W\text{-}\alpha \frac{V_{dw}}{\sqrt{S_{dW}^{Corrected}} + \epsilon}$$

$$b: = b\text{-}\alpha \frac{V_{db}}{\sqrt{S_{db}^{Corrected}} + \epsilon}$$

Adam optimizer updates weights and bias to gain minimum loss. Training completes when loss is minimum.

Interaction between client and bot:

Our system ensures two way communication between clients/users and the bot. The system takes queries from its users as input. After that the input data is being processed and featured to be fed to the model. The data is analysed by the model

and classified under a tag/class. Depending on that the bot responds to the user. Following this cyclic way the interaction works out as described earlier in the project objectives.

5.Experiments

Necessary description for our system is discussed below.

Dataset

The dataset we have used in our system is a json dictionary file with four keys as tag, pattern, response and context. For our training data we took patterns as featured data and tag as classes. There are 100 pattern samples with 21 tags and every tag has 4-10 pattern samples.

Some samples with classes after processing data, are given below.

Pattern	**Tag**
how you could help me	Option
you mean	Hate
what should I call you	Name
you are cool	Like
What your thoughts about it	Thought

As we had a small data, we tested on train data means train and test data are the same.

Evaluation

To evaluate our ChatBot model, we use precision, recall and F1-measure.

- Precision: It is the number of correct positive results divided by the number of positive results predicted by the classifier.

$$Precision = \frac{True\ Positive}{True\ Positive\ + False\ Positive}$$

- Recall: It is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive).

$$Recall = \frac{True\ Positive}{True\ Positive\ + False\ Negative}$$

- F1-measure: F1 Score is the Harmonic Mean between precision and recall. The range for F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances). High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :

$$F1\ measure = 2* \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}}$$

Results

There are 21 tags with more than 100 pattern samples and we get pretty good results. At the time of training for every epoch we calculate loss. To minimize loss we use Adam optimizer. Costs vs number of epoch figure given below.

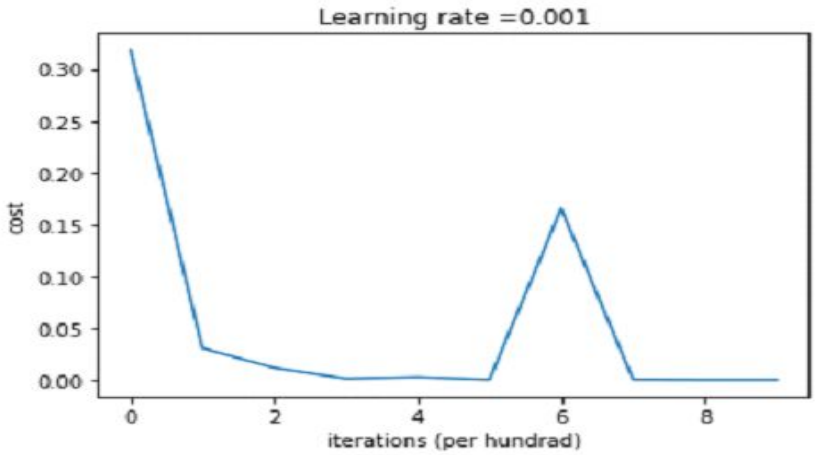

Fig: Cost vs iterations

Confusion matrix for 21 classes (tags):

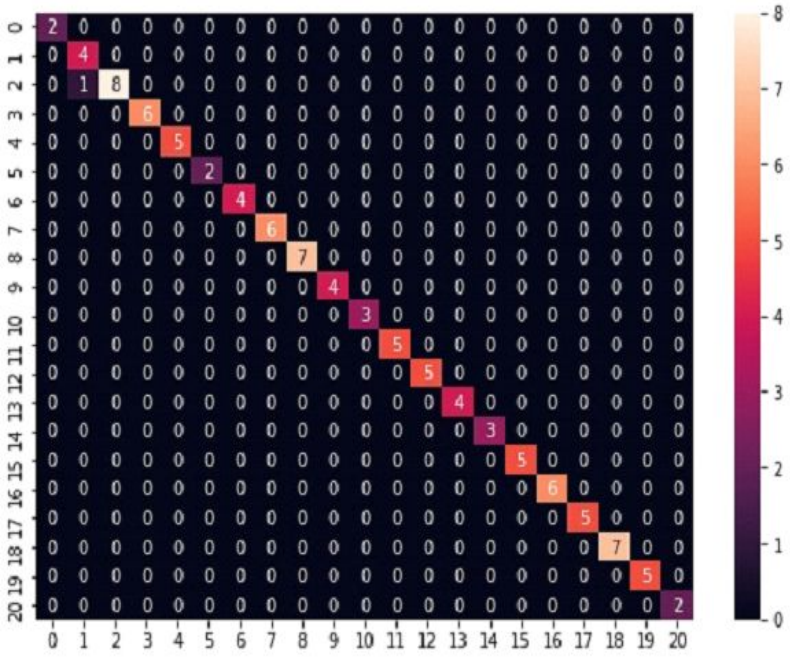

Fig: Confusion Matrix

Precision, recall, F1-measure after training the model:

	precision	recall	f1-score	support
age	1.00	1.00	1.00	2
bad_joke	0.80	1.00	0.89	4
doing_badly	1.00	0.89	0.94	9
doing_great	1.00	1.00	1.00	6
favorite_movie	1.00	1.00	1.00	5
favorite_show	1.00	1.00	1.00	2
good_joke	1.00	1.00	1.00	4
goodbye	1.00	1.00	1.00	6
greeting	1.00	1.00	1.00	7
hate	1.00	1.00	1.00	4
how_are_you	1.00	1.00	1.00	3
joke	1.00	1.00	1.00	5
like	1.00	1.00	1.00	5
name	1.00	1.00	1.00	4
netflix	1.00	1.00	1.00	3
quick_run	1.00	1.00	1.00	5
real_bot	1.00	1.00	1.00	6
sky_net_no	1.00	1.00	1.00	5
sky_net_yes	1.00	1.00	1.00	7
thanks	1.00	1.00	1.00	5
your_thoughts	1.00	1.00	1.00	2
accuracy			0.99	99
macro avg	0.99	0.99	0.99	99
weighted avg	0.99	0.99	0.99	99

Table: Evaluation Matrix

6. Conclusion

For the dataset used in our system to develop, the system shows 100% accuracy. But in case of any large data set the accuracy and the efficiency may deteriorate which is quite normal. To handle large dataset with good enough accuracy AND efficiency the hyperparameters can be tuned. Also the use of dynamic learning rate in each iteration can solve the problem. Overall the model we have used provides satisfactory results. There's also some scope to develop it with more efficiency.

Project Github:

https://github.com/nowshad7/Retrieval-based-ChatBot-using-Neural-Network

Reference:

[1] https://github.com/katanaml/katana-assistant/blob/master/mlbackend/intents.json

[2] (Hu et al., 2014; Ji et al., 2014; Wang et al., 2015; Yan et al., 2016; Wu et al., 2016b; Zhou et al., 2016; Wu et al., 2016a

[3] (Shang et al., 2015; Serban et al., 2015; Vinyals and Le, 2015; Li et al., 2015, 2016; Xing et al.,2016; Serban et al., 2016)

[4] Sequential Matching Network: A New Architecture for Multi-turn Response Selection in Retrieval-Based Chatbots

[5] (Wang et al., 2013; Ji et al., 2014; Wang et al., 2015; Wu et al., 2016b)

[6] Speech and Language Processing. Daniel Jurafsky & James H. Martin. Copyright c 2018. All rights reserved. Draft of September 23, 2018. (https://web.stanford.edu/~jurafsky/slp3/24.pdf)

[7] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.