IMPRINTING MEMORIES AND EXPERIENCES ON NOISE

Tofara Moyo

C.E.O. Mazusa

tofaramoyo@gmail.com

ABSTRACT

We outline a method where we imprint the memory and collective experience of an agent, or the previous words said and in what order and by whom in a chatbot. We are given a training corpus of conversations. Then a feed forward neural network is used to input a noise vector with the current word. At the start of the conversation we will input the zero vector with the dimensions of the noise vector that we will use. Together with the first word found in the conversation. This will output another noise vector and the 1 hot encoded next word. We will then feed the duo output once more into the network to predict the next word. During the learning process we will not alter the noise vectors predicted, but the error function will be parameterized by the word vector. This will marry the two progressions together. That of the noise and the structure of the conversation and which word ought to follow which in a sensible sentence. During operation we input the zero vector and the first word, then input the duo output from that back into the network to predict the next word to be said. When a live response is given to a chatbot the next word will not be predicted, but we shall still get another noise vector with which to feed back in with the next word, until an EOU (end of utterance) token vector is input alongside a noise vector. Then the system begins to predict the words to say as before.

KEYWORDS

Generative Algorithm, Chatbot,

1. INTRODUCTION

This document describes a new way for an AI algorithm to possess memory in the form o marrying two progressions, that of a noise vector and that of elements that the Algorithm must keep track of in the order they come. The inspiration came from the understanding that noise vectors give enormous capabilities to generative algorithms such as GANs [1] and auto encoders, [2] and we should be able to use them to represent more in a principled manner.

2. EMBEDDING MEMORY

2.1. The Algorithm

When we input the first instance of the noise vector and the first word vector, we will get a random noise vector and a random prediction for the word hot code. That is fine, we will simply take that particular noise vector and create our next training case, which will consisted of the found next word in the conversation and the noise vector that was part of the output of the previous training case.

We repeat this till the end of the conversation. In order to mark that the part of the conversation we are currently parsing represents a reply to the initiator, we input an EOU, end of utterance, symbol after the initiator of the conversation has finished. The output to that will be an "R" reply symbol.

When the reply is being given, at each point we predict the R symbol each time we get an input, but we input the next word given by the replier, rather than the R symbol in the next time step. Once we have our training cases we know that, while the noise vector follows a sensible progression (which is meaningless at this moment though fully deterministic)The words predicted don't match the next word that is input with the next word in the next training case.

We will therefore back propagate this error until they in fact do match up. This will introduce the following meaning to the noise vector progression. Since the noise progression stretches out deterministically from the beginning of the conversation, and the noise vectors are paired with words that come from a conversation in their order, we can see the system as imprinting the information of the past of the conversation onto the noise.

During operation we will input words that will give us noise vectors, and we shall use these noise vectors as input alongside the next word in the progression, to predict the next. We in fact actually have a latent space where noise vectors belong. A word vector input alongside a noise vector represents a perturbation of the current noise vector to alter its predictions for the next one. This altered noise vector contains information on which word was said because the neural network processing it can use it to derive the most sensible next word after it (because of the training process), so effectively we are adding information to memory this way.

2.2. Deep Reinforcment

To improve the chatbot we could implement deep Q learning. Human annotators would have labelled different parts of the conversations to represent how they believe the initiator is faring alongside the responder with rewards. These labels will be used as a source of labels for a supervised trained deepQ learning system. At the end of each conversation we will say how well the initiator fared against the responder. Then distribute the rewards across the words in a consistent way according to Q learning theory. We will also have a conversation that represents the initiator being reprimanded multiple times for not saying consistent information. The rewards will be clearly labelled to show an inconsistent reply by the initiator as reducing the rewards drastically. What we would like to do with this process is eventually, given a trained chatbot, trained on multiple conversations with conflicting themes, we could sort of counsel out the insanity that this chatbot exhibits in its responses using the reward structure we would have established while annotating the training conversations and deep Q learning. Understand that we could include factors such as the perceived intelligence of the chatbot when annotating the chatbot, in order for it to increase in intelligence during operation and Deep Q learning

3. Conclusions

We could use this to perform question answering with only a small modification. We shall simply have SOD (start of document) EOD (end of document), SOQ (start of question), EOQ, (end of question), SOA (start of answer), and EOA (end of answer) tokens to include with the noise vector at the appropriate places.

We could also thus predict stock movements based on news articles more effectively, perform document summarization and sentiment analysis this way. We could also generate entire movies and musical pieces similarly using a modified vibrational auto encoder where z represents the noise vector which is being imprinted the history. Also with self-driving cars the cars could be made cognisant of its entire experience, from start of journey till end with this approach.

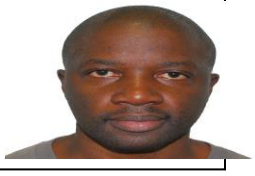
References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio (2014) generative Adversarial nets Authors

[2] *Diederik P Kingma; Welling, Max (2013). "Auto-Encoding Variational Bayes".*

Authors

Photo



Tofara Moyo

Is lead researcher at Mazusa pvt limited.