Maximilian Bragazzi Ihrén maxbr431@student.liu.se

Post project paper

What was your group’s project about?

Our group made a bot based around n-grams, aimed in a general sense towards producing tweets imitating another twitter account. For this project in particular, we decided on Donald Trump, seeing as him and his tweets are a very popular topic, and due to the idea itself seeming fun to do.

After a fair bit of testing which would be best, we went with the choice of trigrams for our bot.

To what extent did your group’s project help you to accomplish the relevant learning objective?

The learning objective in question being: 'judge the difficulty and the feasibility of language technology applications'.

For the actual judging, we decided to conduct a survey of sorts. We made a Google Forms form, where the end users would select (between two tweets) which was actually tweeted by Trump, and which one was made by our bot. From the results of this, we deducted that even though the majority of people weren't fooled for the most part, it was still very close. An estimation of the total correctness considering all questions would be that about 55-60% of the users chose the right tweet, which in turn means we fooled people during 40-45% of the answers, which is a fairly good result given that over 50 people answered the survey.

However, this data might be skewed for a few reasons:

1. **Choosing what tweets to display**

   Due to the fickle nature of n-grams, we had to choose which tweets we would display to the end user, since a lot of the tweets the bot generated weren't coherent enough to be able to fool anyone.

2. **Participants may have not read the instructions**

   The instructions said to try to guess which tweet out of the two per question was the tweet that the bot generated. While we have no reason to believe someone didn't read the instructions things like that often happen, and people might have tried to choose what tweet they thought was from the real Trump. If this is true however, it was most likely a small portion of the participants.

3. **Participants may not have been completely serious when answering**

   Due to the humorous side of the project, some people might not have been completely serious when they answered the survey. For example, some people might have chosen the tweet they thought would be less funny that the real Trump would have tweeted for what they thought the bot had generated, just because they wanted to believe that Trump was

more ridiculous than he is. One tweet that we generated was for example that Meryl Streep was part of the Soviet Union, which would seem a bit far fetched, but it would definitely be funny if Trump actually had tweeted it. That particular tweet got about 30% percent of votes in its question, which to me points to the fact that people might not have been fully serious.

What knowledge and what abilities were you personally able to contribute to the project?

As our group consisted of 6 people split 50/50 across cognitive science students and innovative programming (IP) students, each individual didn't get to bring a lot of uniqueness to the project. Instead, each of the halves brought their expertise when needed in order to maximize efficiency.

For instance, us IP students were largely responsible for the first half of the project; collecting the data from twitter, and building the n-gram model that would later produce the tweets. The cognitive science students came in more in the latter half, and fixed a lot of the things that us IP students haven't had a lot of practice in during our education, such as the actual form the users had to fill in, and the layout and structure of the presentation.

In terms of what I personally contributed (within the IP part of the group), I worked a fair bit together with the rest to build the program that collected twitter data. The actual n-gram model had most of the base built by one of the other IP students during his spare time, so there wasn't much we could do to help there. However, when it came to refining the model in order to fit the task at hand, we all helped in our own little ways in order to make it better.

In the IP program we've had a very big focus on pair (and even group) programming, which is why I think there isn't much to be said about the individual contributions of the group members. We often sat together in front of the same screen, discussing how to move forward or make things better, implementing those changes, etc. This leads to the individual contributions being very (for the lack of a better word) muddy.

Based on your group’s project, how would you judge the difficulty and feasibility of the application that you investigated?

As mentioned above, we did manage to fool a large part of our participants. This would in general mean that using the n-gram model to generate sentences that replicate ones written by real people is feasible. However, that statement comes with a big **but**.

As also mentioned above, we did have to select which tweets to test our participants on. This is largely due to the fact that n-grams can produce very coherent sentences, but it can also produce ones that definitely are not. N-grams in themselves do not have any form of sentence cohesion check, such as the generated text follows a proper sentence structure or anything like so. This can lead to generated sentences that (for example) change subject halfway through the text, or simply don't follow any structure and end up incoherent.

Problems like these can of course be fixed by increasing the order of the n-gram, but this also leads to more and more generated texts being copied directly from the training data. For tweets, which are limited to 140 characters, we found that quadgrams simply copied the training data about once every third iteration, which is far too much. Trigrams (which is what we ended up using) only copied a tweet about four times out of all the tweets we generated using it (which was probably in the 100s). However, since we only look back two words in order to determine the next, the model can be very incoherent at times. Another minor factor into why you shouldn't increase the order for this particular project is that the character limit for tweets favors smaller sentences, which are better formed by smaller orders. A 71-gram would very rarely be able to produce something below 140 characters, for example (and if it did it would most likely not be coherent, since all the words would have to be 1 character long).

Another limitation was the fact that we didn't limit the bot to generating tweets with a maximum length of 140 characters. This was mostly due to time limits, since the best we could have done given the time frame is to add a check every time a new word is added, and if it exceeds the length send out the text as finished, even if it cuts off mid-sentence, which naturally isn't ideal.

A model we could have gone with is to increase the probability of the end of sentence tags the longer the sentence got, so that they culminated at 100% at 140 characters, but this would require a lot more work on our part that we simply didn't have time for. This might of course also generate sentences that cut off mid-sentence, but it would also increase the chance of the bot stopping at, say, 120 characters, where it might have a coherent flow.

Another way we could solve this problem that we discussed was that you could create a model on top of the n-gram model that takes in a context free grammar ruleset, and generates a sentence structure from that. The n-gram model would then generate the next words like normal, but with the added criteria of the next word needing to be tagged with the next tag in the generated structure. This would hopefully make the models generated texts more coherent.

You could, on the other hand, argue that we would be altering the n-gram model to fit our standards, and it would therefore no longer be classified as n-grams (n-gram++?). This would take away from the question we wanted to answer, which was whether or not n-grams are viable for this sort of project (and to check if n-grams are a good model over all).

So, to conclude, n-grams are a very basic model to use for text generation, but with the right filters (both for the training data and for the end result), it is certainly a good enough model to use for such a simple project, which can be deduced from the end result of our survey.