DAT410/DIT728

Group 54
Module 4: Natural Language Processing

Jonatan Hellgren, 960727-7010
Matematikprogrammet GU
gusheljot@student.gu.se

Ankita Rahavachari, 19960709-4605
MPDSC Chalmers University of Technology
ankita@student.chalmers.se

15th February 2021

We hereby declare that we have both actively participated in solving every exercise. All solutions are entirely our own work, without having taken part of other solutions.

Hours spent working

Jonatan	Ankita Rahavachari
20	15

1 Reading and reflection

1.1 Jonatan

1.1.1 (a)

I believe that most of the old school AI problems that is still relevant today as this wide range of approaches, since they have lived through the innovations in the technology and the field. Some examples of such subject are: game playing AI for games such as chess, path finding for gps and automating math proofs.

1.1.2 (b)

They are essentially the same thing, a function that takes as input a sentence from one language and outputs a translated sentence in another language. However how they do it is very different, the rule-based systems are written with human intuition while statistical methods are based on probabilities and neural methods is basically a very complicated function optimized with gradient descent.

1.1.3 (c)

When the predictability of a system is the top priority and the domain is small, then a rule-based system would probably perform better, basically when the whole domain can be mapped by programmers and linguists. For example in online web application like banking or shopping, here the domain can be small enough to make it entirely mapable.

1.2 Ankita Rahavachari

1.2.1 (a)

In general, all technologies involving Artificial Intelligence have undergone different phases of improvement as it gets updated based on new ideologies. One example where we can see a wide range of approaches is in the field of robotic process automation. It is one field in which there is constant change in the use of underlying technologies to match the present requirement.

1.2.2 (b)

Both rule-based translation system and neural system work for the same objective i.e., to translate the given file to the target language. The similarity that these two methods hold is that they allow the words to have different meanings depending upon their context and placement. Unlike the traditional decoders they work well with contextual references. The working of both models is however different. The rule-based translation model takes into account the grammar rules and dictionary meaning while translating and the latter is more advanced and it doesn't consider the rules.

1.2.3 (c)

The old school rule based systems can be used when there is strict inclusion of rules and when we are comparatively dealing with a small data set. They are very consistent and they produce good quality predictions. For example: Translators used in day to day web applications.

2 Implementation

In this part of the assignment we will implement the statistical translation model from IBM called: model 1. The corpus which it is trained on is from the Europe parliaments. We have chosen to translate from Swedish to English.

2.1 Warm-up

In this part of the assignment the coding was trivial, no major assumptions where made except what characters will be removed in the training corpus. The signs that where not included is the following: (',', '.', '-', '(', ')', '!', '?'). In retrospect more could have been removed, for example numbers.

In table 1 we can see the top ten occurring words in both the Swedish and the English texts.

Word	Occurrences	Word	Occurrences
att	9181	the	19322
och	7038	of	9312
i	5949	to	8801
det	5687	and	6946
som	5028	in	6090
för	4959	is	4400
av	4013	that	4357
är	3840	a	4269
en	3724	we	3223
vi	3211	this	3222

Table 1: Top 10 Swedish and English words

By counting the number of times the given word have occurred with `Counter` and then dividing it by the total number of words we get the following probabilities:

$$p(\text{speaker}) = 10/256365 = 3.9 \cdot 10^{-5}$$
$$p(\text{zebra}) = 0/256365 = 0$$

2.2 Language modelling

The implementation of the language model i.e. the bigram-probability computation was as well a quite trivial one. We take as input to words then we look at all the times the first given word has occurred and store the distribution of the following words, after this we compute the probability that the second given word will occur afterwards. Later we multiply the probability of each set of words in a sentence. Since the training corpus doesn't contain all pair like this, this probability is likely to become zero, thus we do not actually multiply with zero, instead we multiply with the largest number of the probability and $1 \cdot 10^{-7}$, this number was arbitrarily chosen.

So for example if we compute the probability for the sentence 'I think therefore I am' we would get the following probability:

$$P_{bigram}(\text{I think therefore I am}) = 1.05 \cdot 10^{-7},$$

but if we add the word zebra to the end of that sentence we get the probability:

$$p_{bigram}(\text{I think therefore I am zebra}) = 1.05 \cdot 10^{-13},$$

the second sentence is one million times more unlikely than the first one, this is since the word zebra does not occur in the corpus.

2.3 Translation modelling

By looking at the column in the t-matrix that corresponds to the word European, we can check which Swedish words has the highest probability. The words with the highest probability is the following in descending order:

- europeiska
- europeisk
- europaparlamentet
- i
- att
- unionen
- till
- och
- de
- den

so with this word the model managed to find the right translation. Note that the conditional probability for 'europeiska' given 'european' was:

$$p(\text{europeiska}|\text{european}) = 0.883,$$

so it was also quite certain of this translation. There are some nonsesical words in this list, but after the word 'i' they all have probability less then 1% and is probably just in the list because they are common words and are thus likely to show up in the same sentence as 'european'.

2.4 Decoding

In this part we are going to find the English sentence E^* that satisfies the following expression:

$$E^* = \text{argmax}_E P(E|F),$$

to do this we would in theory have to try all possible translations then take the product of all the probabilities for each of the English words then multiply that product with the bigram-probability of the English sentence. For an english sentence E and a Swedish sentence S, the formula looks the following way:

$$\left(\prod_i^n P(E_i|S_i)\right) P_{bigram}(E),$$

where the index i refers to the ith word in the sentence.

However doing this computation for every possible translation isn't feasible since there are 11095 Enlgish words in the corpus and for a sentence of length n that would become 11095^n possible permutations of sentences. So what we implemented to solve this issue is that we only look at the top 3 contenders for each Swedish word, with this we have shrunken the possible sentences to be 3^n. Since a word that isn't in the top 3 most likely is probably not a good candidate anyways this is a reasonable assumption.

To see how well the model performed we can take a look at some examples in Table 2. As we can see the model didn't perform that well, it managed to find some words that corresponds to each other but it would pass as proper English. This is however to be expected when using such a simple model as IBM 1, the more advanced ones would probably be able to translate this more accurately.

Swedish sentence	Translated English sentence
Jag tror därför är jag	I belive therefore is I
Herr talman Europa behöver mer mat	Mr President Europe more eat
Jag är inte glad	I is not glad

Table 2: Some examples of translations

3 Discussion

3.1 (a)

For a translation to be good, it should maintain the fluency and proper contextual relationship between the words in the sentence.

Manual procedure: For evaluating the translation is through a human translator who knows both the original and translated language. A human translator can understand the flow of a sentence and determine if the translation is contextually meaningful.
Disadvantage of this process will be that the humans might miss some tiniest changes and it is also hard to translate/evaluate a lengthy document.

Automatic procedure: Before deployment of the ML model, A/B testing and experimentation can show how well the machine learning model is able to translate, enable us to understand the underlying data sets and help us to focus on the flaws.
Drawback of A/B testing is that it can take up more time to set up than other forms of testing.

3.2 (b)

Based on the data present in the training set these translations can be a feature because the machine learning system has translated every sentence based on its learning using the training data set. There might have been male doctors and female babysitters in the training data set but if that is not the case then this is considered as a bug because there is no explicit mention of names to identify the gender to relate to a particular work category as both male and female can be a babysitter or a doctor. So when there is presence of nouns in a sentence we think there should be a clear relation to the gender as well.

3.3 (c)

In the first two cases it looks like the translator has achieved to translate the sentence based on contextual relation. For example, in the first sentence "I hit the ball with a bat", the word hit, ball, and with a bat has a meaningful relationship with respect to the context. Therefore the final translation also has proper relationship between the words in the sentence.

On the other hand for the third sentence, the translator has simply decoded every word to its respective Swedish word instead of following the contextual meaning. This leads to the addition of unwanted words and makes the final translation meaningless.

4 Summary and Reflection

4.1 Jonatan

4.1.1 Summary of Lecture: Natural Language Processing

Natural langueage processing can be used in spamfilters, spellcheckar, machine translations and dialogue systems for example, but is ofcouse used in a wider variety of fields aswell. Anywhere when we are trying to train a model where that data is language we are in the natural language processing field. Working with language as data can be hard since it becomes very sparse and many words wont even be collected, and a majority of the collected words will be common ones like the, is and I.

It can be quite difficult for a model to find underlying patterns in noisy data such as language, to help the models out tagging is sometimes used. With tagging a human tags certain parts of the text that the model should look for. Other techniques used for helping the models out are parsing where we basically clean up the data a bit.

NLPs history started in the 1960s with mostly rule-based solutions, it didn't however work out that well since language turned out to be a bit more complicated to model then expected. In the late 80s the comunity started to focus on data-driven models and later went on with probabilistic ones like the IBM model 1 in the 90s. Then in the 2000s linear models started to become more popular, which where later taken over by neural models by 2015. So there have been a wide range of methods for this task.

4.1.2 Reflection on Previous Module: AI Tools

In the previous module we learned how to use AI tools in the proper way, the tools are basically code but very specified off course. A big emphasis was on the structure how we construct models, the fit, predict score paradigm used in scikit-learn was recommended when coding regression or classification models.

4.2 Ankita Rahavachari

4.2.1 Summary of Lecture: Natural Language Processing

Natural Language Processing is widely used in different applications such as spam filters, spell checker and grammar checkers etc. Language is typically different from other types of data as they are sparse, diverse and discrete. During the early days the dominant NLP system was based on Rule-Based Translations and the present is an advanced Neural model. The advantages of using a Neural model is that they are more expensive, provide end-to-end feasible solutions. Linear Models are still a baseline for many NLP tasks as they are easy and fast to train.

Difference between machine translation and interlingua translation is that the machine translation translates the sources directly to the target language but the latter maps the source language to a meaningful representation and then converts this representation to the target language.

The data driven machine translation system is trained on an example text. Example: Word-Based and Phrase-Based statistical systems and neural systems. The Neural Systems has an encoder to summarize the information in the source text and a decoder to generate a target language output based on the encoding. Drawbacks of this model is, it requires tweaking of data and consumes lots of energy. The training time is also more when compared to other models.

In probabilistic machine translation we learned to decompose the probability by rewriting it as $P(F \mid E)$ and P(E). By doing this we achieve the right content, can take care of fluency and also training becomes easier. Here $P(F \mid E)$ is known as the translation model and P(E) is called the language model.

Markov's assumption states that the next word depends on the current word and not on the history. From this assumption the Bigram model was formed. Later to make the probability of the translation model less complicated independence assumptions were introduced.

Expectation-Maximization algorithms solve the problem of estimating parameters when some parts of the model are latent. E-step computes "soft counts" based on expected values of latent variables M-step re-estimate parameters, based on the "soft counts".

4.2.2 Reflection on Previous Module: AI Tools

In the first part of the module, the paper gave an understanding of how an ML engineer should be aware of the design choices they make and the model should be designed in such a way that it can be subjected to future changes without incurring massive cost in the future analysis and during maintenance. The implementation of K-NN classifier from scratch gave an in-depth knowledge of working on the fit, predict, score model and we were also able to analyse how well the system worked. In the discussion part of the assignment we were able to note the assumptions that the developers make during the training process of the machine learning system and possible impacts due to the same. We came to a conclusion that an ML engineer is engaged with the model they develop even after the deployment in-order to follow up with their clients to check the performance of the model and maintain it in the right manner.