NLP in the Swedish Medical Language Data Lab

A summary of models and methods used in the NLP project.
By Markus Sagen, Peltarion and Olof Mogren, RISE.
November 2021.

What is NLP?
Natural language processing (NLP) is the study of how to teach computers to understand natural human languages programmatically. It is a cross between linguistics, computer science, and machine learning and combines rule-based systems from linguistics and statistical learning approaches from statistics, machine learning, and deep learning.

Computers represent all things as numbers. Images, therefore, have a natural way to be represented in a computer, where each pixel of an image can be represented as the color intensity of red, green, and blue. Objects in an image are groups of pixels, and pixels within an object are likely to have similar color values. This means that images, objects, or even pixels can be measured and compared by a computer.

For text, there are multiple different ways of representing characters and it can be much harder, since words are symbolic, and there is no natural way of measuring similarity between such discrete objects, for instance:

- Changing a letter in a word makes it a wholly different one:
  Ex: "*friend*", "*fiend*", "fend", "lend"
- The same word can have a different meaning.
  Ex: "*saw*", "*saw*"
- Words can be negated to have the opposite meaning
- Changing word order can give a drastically different tone, etc.

For this reason, words need to be represented numerically, ideally to capture the complex linguistic characteristics needed to understand languages. The way this representation is done for training machine learning models for text is by creating so-called *word embeddings* and then using these *word embeddings as input* to machine learning models. These word embeddings are created in such a way to represent the different words numerically but also capture the semantics. Word embeddings can be either *Sparse* or *Dense* and enable operations, such as comparing word similarity.

But, before a word embedding is created, the text first needs to be split into words or parts of words. This process is called *tokenization*, and the words or sub-words extracted are called *tokens*.

The difference Between Dense and Sparse models and their representation

Models trained using dense or sparse word embeddings as input are sometimes referred to as *Sparse* or *Dense models*. As a general rule, sparse word embeddings count the occurrence of words or groups of words, whereas dense word embeddings are learned from machine learning models.

Bag-of-words is a way of counting the frequency of words in a document. This count of the words can either be for one word at a time or groups of words, so-called n-grams. **TF-IDF** is a technique to take into account which words have the most importance across documents. Stopwords, such as “a” and “the”, may be very frequently used in multiple documents, and therefore be less important to a specific document; whereas words that only appear in a few documents will be viewed as more important. *If certain specific words are key for solving a problem*, then these methods are preferred.

Transformer models, which have been used predominantly for solving NLP problems since their introduction in 2017, are deep learning models that create their dense word embeddings, which are learned and stored in the network. By using dense word embeddings, and using deep learning models for training allows Transformer based models such as BERT, GPT, T5, etc. to learn more complex linguistic relations. If the model needs complex language understanding, such as sentiment analysis, question-answering, sarcasm detection, then these methods are preferred. To learn and understand languages, Transformer based models are usually pre-trained on a massive unlabeled dataset initially to learn general features of a language before being fine-tuned to solve a specific problem.

The potential risk of extracting words from the models

Machine learning models that have been trained using a separate word embedding, have only learned to use the frequency of the words, but haven't seen or learned what the words “actually are”. This potentially means that if the word embedding is removed and separated from the model, the trained models can still be shared, but without knowing anything about the data it was originally trained on. But, in the case of Transformer models, such as BERT, the word embedding is encoded into the model and parts of the data, or even sentences could potentially be recovered, though noisy.

A simple bag-of-words model combined with a linear classifier can leak information about word frequencies, but not of word order. If the vocabulary is defined using non-sensitive text, one can also guarantee that sensitive tokens will not be included in the vocabulary part of the model.

The models we have used

We have used both dense and sparse models in the SMS project for different aspects, for two primary stakeholders: *Folktandvården Västra Götaland* and the *Region of Halland*.

Folktandvården Västra Götaland

Their use-case was to understand in which instances dentists were prescribing antibiotics and what they based their decisions on, based on the journal text of the patient. The goal was to identify the cases where antibiotics should not have been prescribed, and how they could be used in the future to reduce incorrect prescriptions.

We used Transformer based models for this task because the models would need to reason about the context of the patient journal notes, made by the dentists. Since the text was medical text in Swedish, we tested several models:

Transformer models pre-trained on only Swedish text:

KB-BERT
KB-ALBERT
KB-ELECTRA

Transformers pre-trained in several languages:

Multilingual BERT (mBERT)
XLM-R
mT5

We also used a Swedish BERT model to detect named entities, such as person names, regional locations, etc. to anonymize the data. The model used for this was:
KB-BERT for NER

Region Halland

Their use-case was primarily two aspects: if it was possible to find which patients have had a fall injury, based on past medical journal entries; The other was if it is possible to predict future fall injuries, based on the journal text and other features in the electronic health records (EHRs).

Identifying past fall injuries (the first use-case)

Since the task is to identify when a fall injury has happened, then the words used by the doctor are highly probable to contain certain keywords when noting it in the journals, words or n-grams such as "fallen", "on the floor", etc. This meant that sparse models were believed to perform well, and found it to be true for those and BERT. We tested the following models:

- **Sparse**
    - Bag-of-words representation with a linear classifier
    - TF-IDF representation with a linear classifier
    - TF-IDF representation with a random forest classifier
    - TF-IDF representation with a support vector classifier (SVC)
- **Dense**
    - KB-BERT

Predicting future fall injuries (the second use-case)

We started recently and all the models that we would like to train and use for the task have not yet been decided upon. We have so far approached solving the problem by either:

- Training a classification model on past journal texts for each patient and predicting risk of future fall injury for each patient
    - **Sparse Model**
        - TF-IDF representation with a support vector classifier (SVC)

- Train a tabular model on all features in the electronic health record (EHR), except for the journal text. The features used include patient age, gender, hours at the hospital, etc.

    - **Dense**
        - TabNet, but since no journal text is used, it does not need a word embedding.

Next step

The work done so far in the project has shown great potential for how AI-based solutions could greatly impact and aid medical professionals in their effort to provide precision health care for patients. A large part of the project has been learning how to work with sensitive data, getting access, setting up legal frameworks, and working within the realms of GDPR, which have been crucial learnings. This also meant that model training has not been as extensively developed as we initially planned. The following presents potential next steps for future projects.

AI in production

The end goal for all parties involved in the project is to implement usable and practical AI systems that are integrated into existing systems. This includes metrics and methods for monitoring the model performance, dealing with noisy data, outlier detection, using the model in low-resource systems, scaling, etc. Models in deployment may also need to be re-trained on new data, which means that models and datasets should be versioned. To compare the performance of different models or improvements in models, multiple models may need to be run at the same time and compared using A/B tests.

Multi-modal models

Much of the project has focused on what AI models can learn from journal entries in the form of free text, but in health care, many attributes are collected during a patient's stay. To make better predictions, we want to leverage all available data for a patient, which is often in multiple different modalities, such as ECG, lab test results and other categorical features, time-series data, free text, and more.

Previously, most multimodal models have been complex hybrid models composed by domain experts and extremely hard to interpret or modify - one model for one modality which are merged together. However, in the last two years, new models have been presented that show promise to deal with multiple modalities at the same time in one model. The potential for these models is huge because they can learn the relationship between different modalities directly and they can be generalized to new tasks without feature engineering by domain experts. In addition, since these models operate on the data directly and are transformer-based, they are more interpretable.

Enabling data access

Because of the sensitive nature of the data and the necessary legal restrictions on how it can and should be accessed, we believe that finding ways of training on the data without the need to move it from its original location would be beneficial. We believe that interesting avenues for this could be using federated learning and training on synthetic data.

Federated learning enables models to be trained on decentralized or smaller devices with local data, in a way that ensures that the data stays on that device. This is something that several partners in the project have expressed an interest in, primarily training models across servers in different regions and combining the learnings into one model, since sharing the data between those servers can be difficult.

Another important step for enabling working with data-driven methods as those discussed in this project are methods for getting started easily making data available, and the processes, and infrastructures around it. Apart from the learnings from this project and the framework of data readiness levels (Lawrence, 2017, walk-through from AI Sweden here), we also see great potential in providing best practices, guidelines, and tutorials on how to assess data readiness, as well as the legal and practical infrastructure around it.

Interpretable prediction

For any critical system where AI is used for decision-making, there must also be a level of insight into what those decisions are based on and how they are made to build trust in those predictions. The field of explainable AI aims to interpret what these models are basing their decisions on, with methods such as Integrated Gradient and SHAP. An issue with some current explainability methods is that they merely highlight what in the data was used for a certain prediction, but not why.

More importantly, explainability is not merely highlighting the data used, but presenting it in such a way that it becomes useful, practical, and interpretable by the end-user. Achieving this is both an AI and UX problem that depends on the data modality, task, etc. Ideally, end users should be able to influence the models learning and training by correcting the explanations the model provides. This would enable model calibration and inhibit trust and understanding from domain experts. However, this is still an open research question on how to best implement and impact model training in such a way.