Question Answering and Chatbots

3rd Practical exercise – Question Classification

Aleksandr Perevalov

aleksandr.perevalov@hs-anhalt.de

October 4, 2021

Hochschule Anhalt

Anhalt University of Applied Sciences

Text Classification Task

Text Classification Task

Having **a sequence of text documents** $D = [d_1, d_2, ...d_i]$ each document d_i has a **class** c_i assigned to it (**training data**).

Text Classification Task

Having **a sequence of text documents** $D = [d_1, d_2, ...d_i]$ each document d_i has a **class** c_i assigned to it (**training data**).

A classification algorithm has to be trained, such that it can learn to predict document classes based on the given data as precise as possible.

Text Classification Task

Having **a sequence of text documents** $D = [d_1, d_2, ...d_i]$ each document d_i has a **class** c_i assigned to it (**training data**).

A classification algorithm has to be trained, such that it can learn to predict document classes based on the given data as precise as possible.

The classifier has to predict classes not only for the training data but for previously unseen data.

Text Classification Task

Having **a sequence of text documents** $D = [d_1, d_2, ...d_i]$ each document d_i has a **class** c_i assigned to it (**training data**).

A classification algorithm has to be trained, such that it can learn to predict document classes based on the given data as precise as possible.

The classifier has to predict classes not only for the training data but for previously unseen data.

Types of classification:

Text Classification Task

Having **a sequence of text documents** $D = [d_1, d_2, ...d_i]$ each document d_i has a **class** c_i assigned to it (**training data**).

A classification algorithm has to be trained, such that it can learn to predict document classes based on the given data as precise as possible.

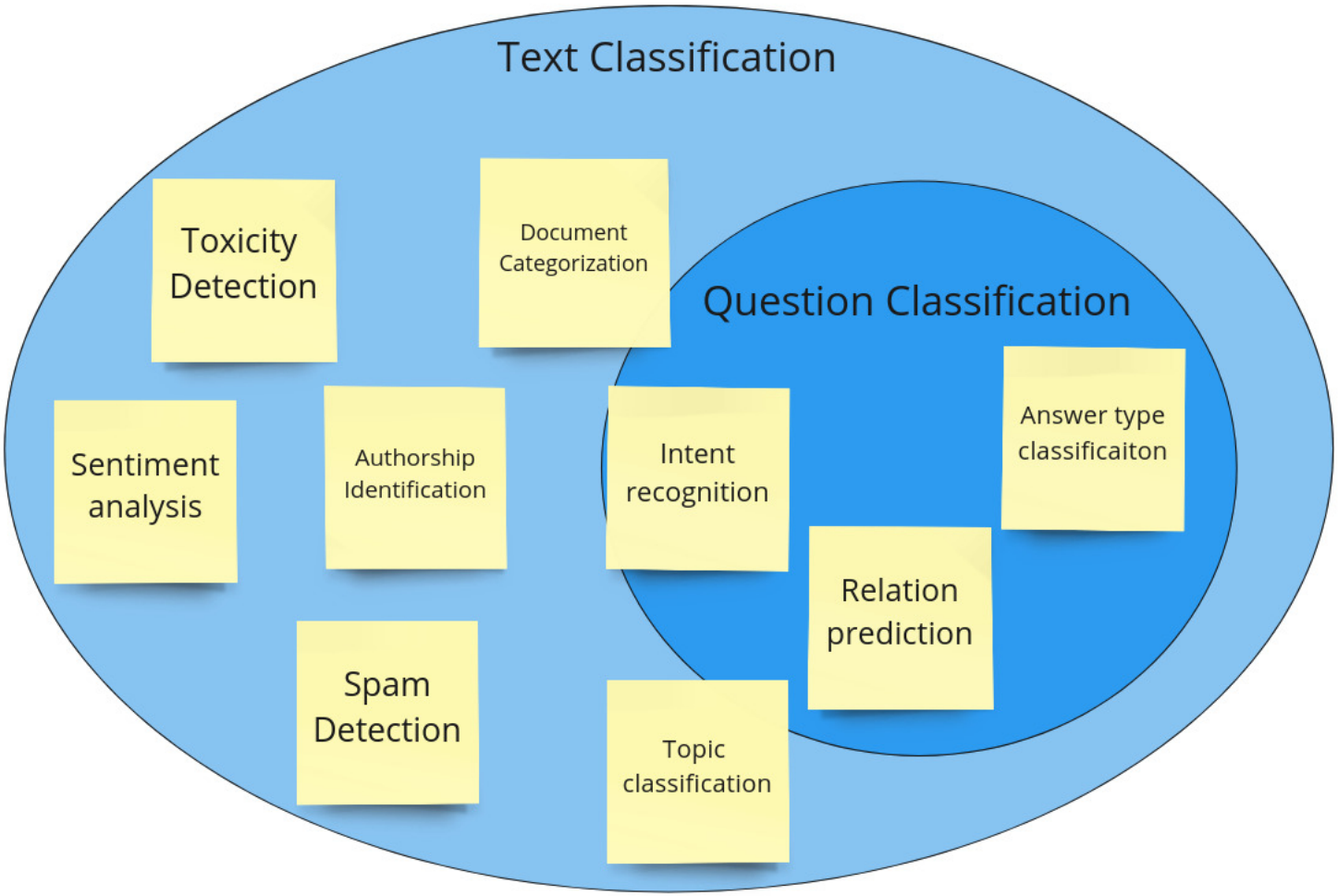
The classifier has to predict classes not only for the training data but for previously unseen data.

Types of classification:

Binary	2 classes
Multi-class	$>$ 2 classes (typically)
Multi-label	a data item might have $\geq$ 1 class

Text or Question classification

Text or Question classification



Question classification in QA

Question classification in QA

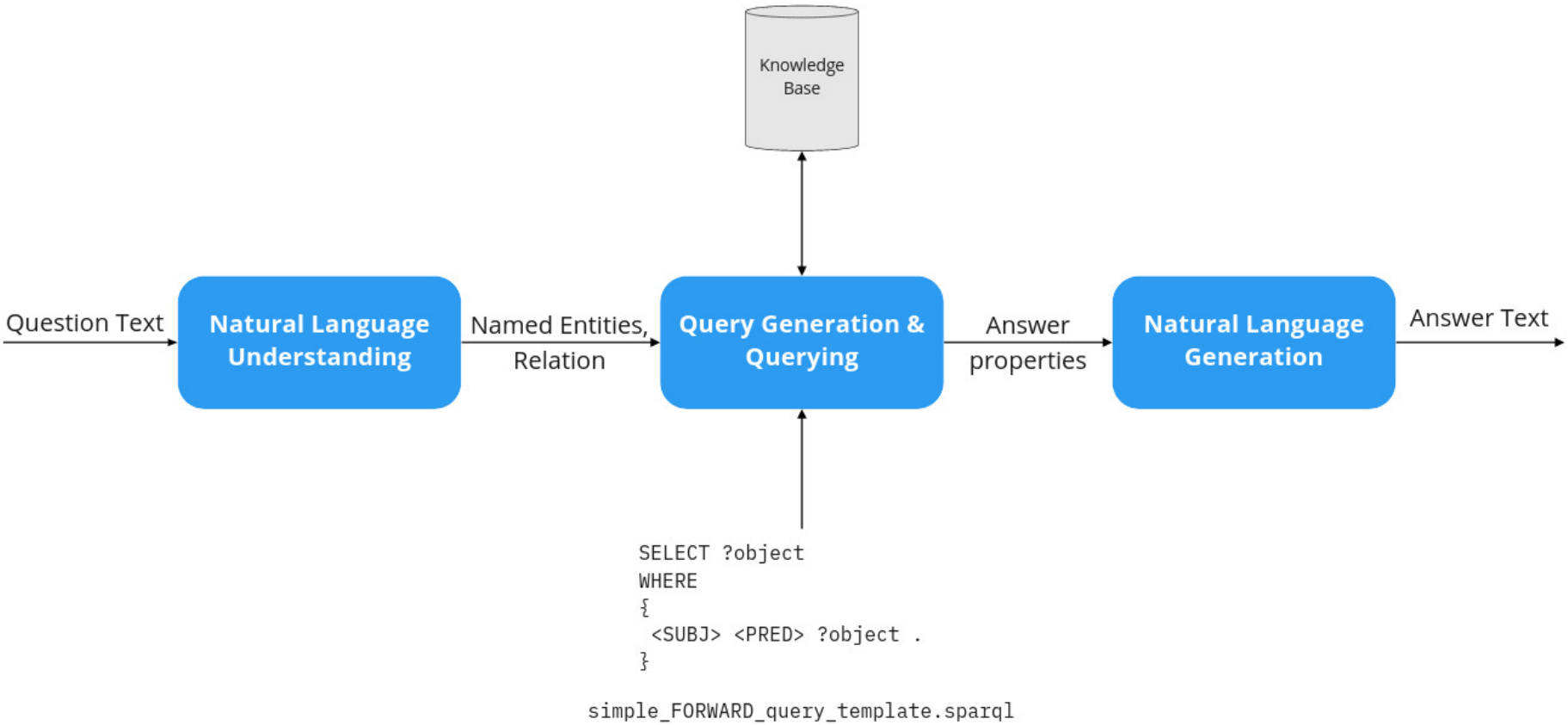


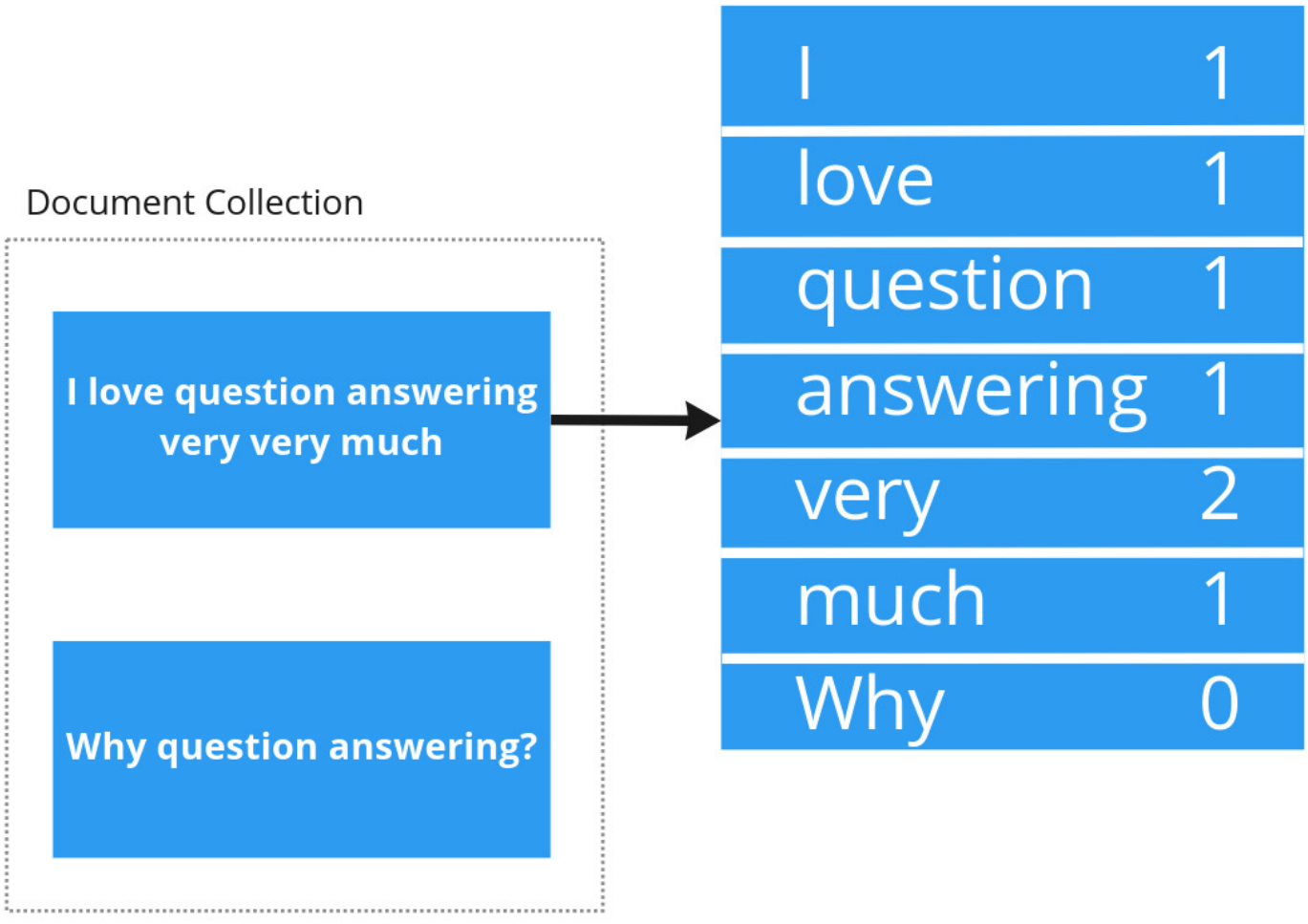
Figure: QA system architecture for "Simple Questions"

Any questions?

Text-to-vector transformation

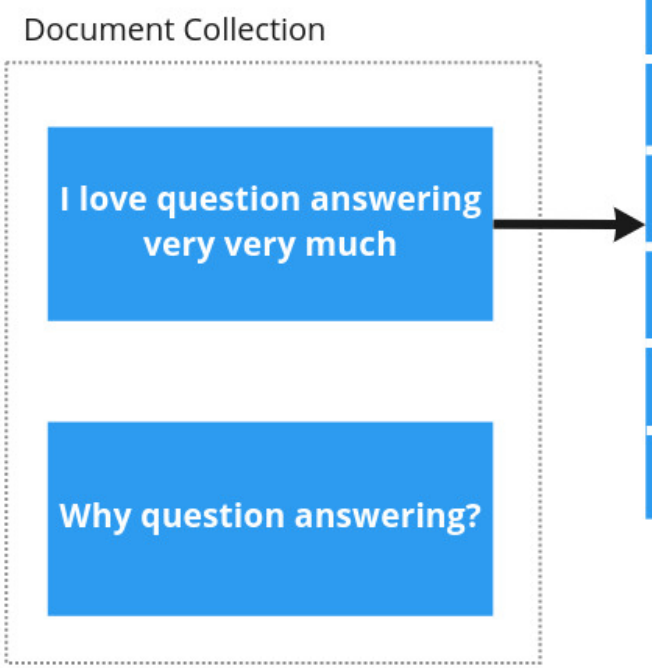
Text-to-vector transformation

Bag of Words



Text-to-vector transformation

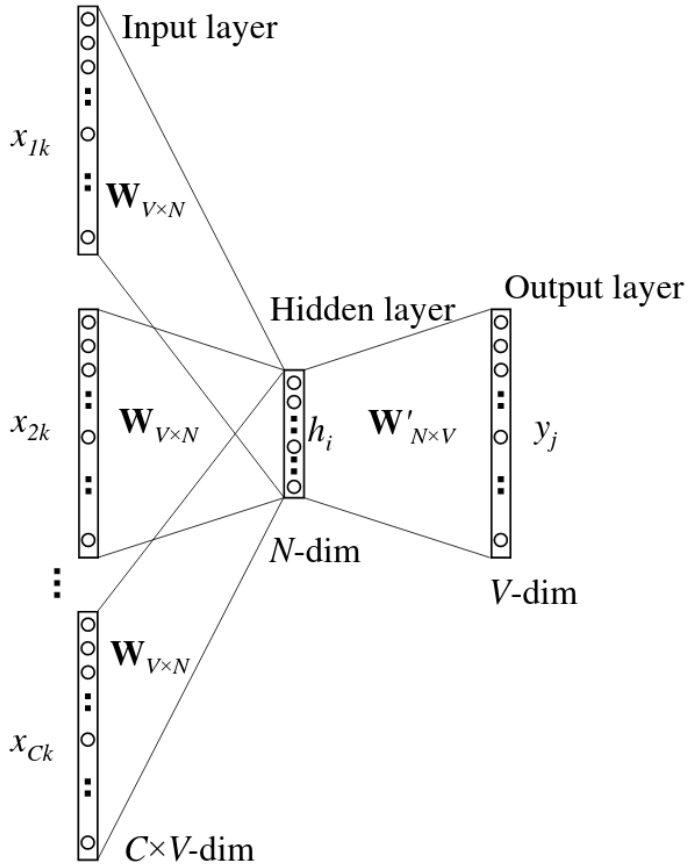
Term Frequency - Inverse Document Frequency (TF-IDF)



	TF	IDF	TFxIDF
I	1/7	log(2/1)	0.099
love	1/7	log(2/1)	0.099
question	1/7	log(2/2)	0.0
answering	1/7	log(2/2)	0.0
very	2/7	log(2/1)	0.198
much	1/7	log(2/1)	0.099
Why	0/7	log(2/1)	0.0

Text-to-vector transformation

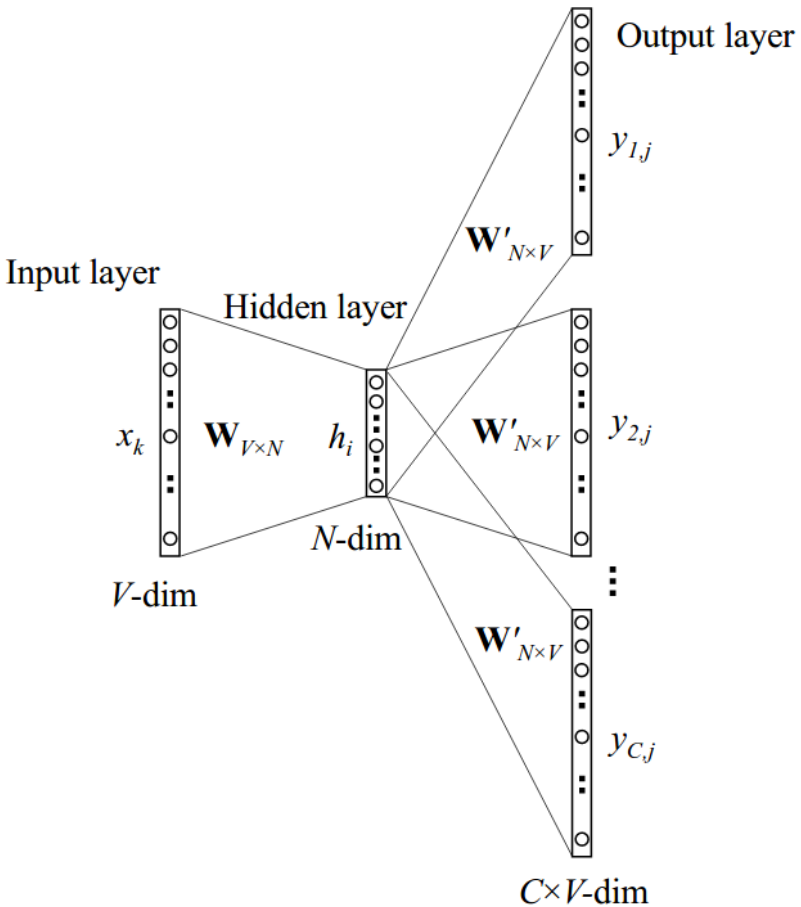
Word2Vec (2013) – Continious Bag of Words [1]



[1] `https://arxiv.org/pdf/1411.2738.pdf`

Text-to-vector transformation

Word2Vec (2013) – Skip-Gram [1]



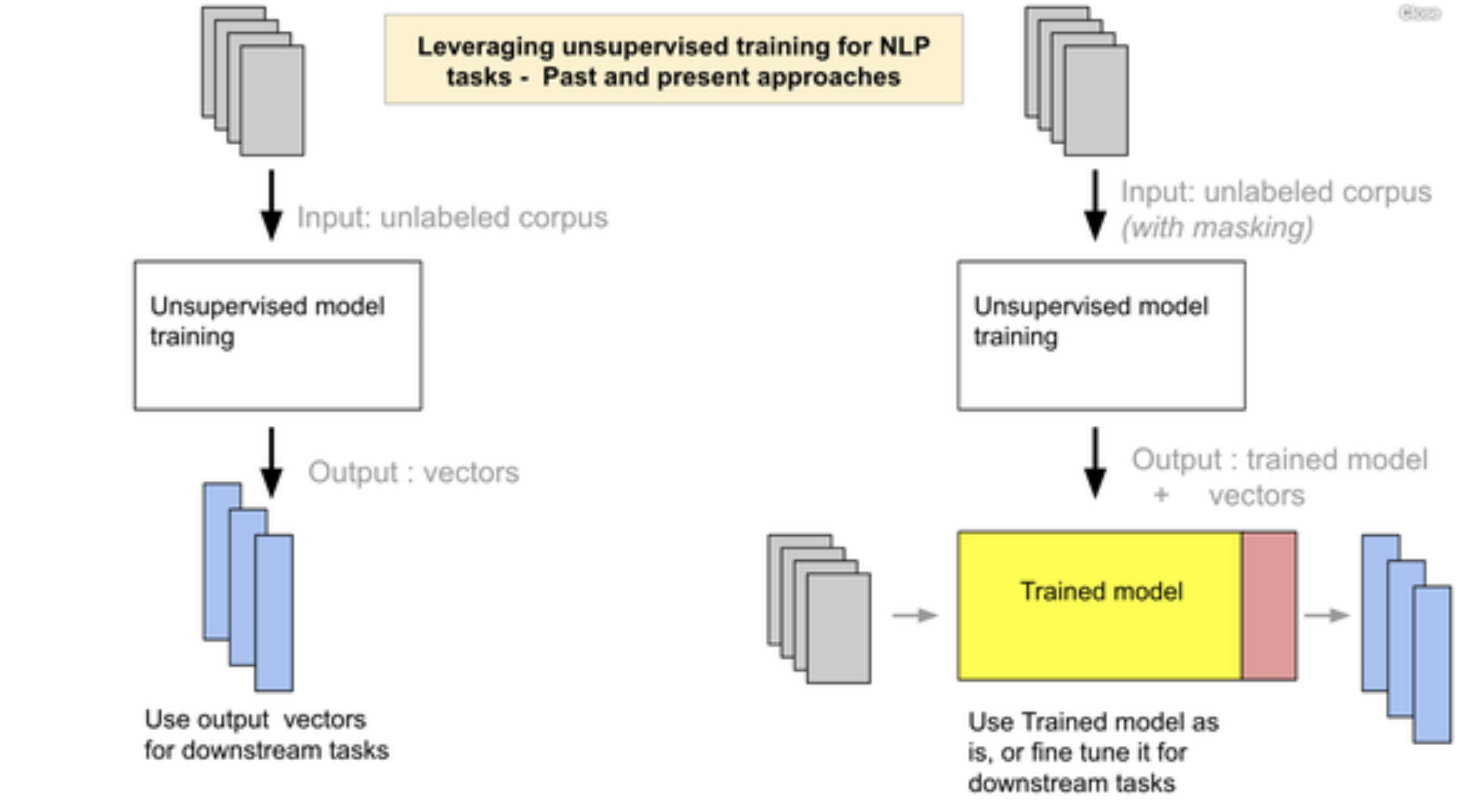
[1] https://arxiv.org/pdf/1411.2738.pdf

Text-to-vector transformation

fastText (2016) – Same as Word2Vec, but instead of words character n-grams are considered as an input.

For example the word vector “apple” is a sum of the vectors of the n-grams “ap”, “app”, ”appl”, ”apple”, ”apple”, “ppl”, “pple”, ”pple”, “ple”, ”ple”, ”le” (assuming hyperparameters for smallest ngram is 3 and largest ngram is 6).

Modern approaches



[2] https://www.quora.com/What-were-the-most-significant-Natural-Language-Processing-advances-in-2018

Let's do the exercise. Ask me if you have a question.

Plan for the Exercise 4: Back-end and Front-end

Task for TODAY: implement a Back-end and/or Front-end parts.

Back-end – is a Web-service which works as an API and implements 2 methods:

1. Type: GET, Name: health, Returns: "Hello World" string;
2. Type: POST, Name: get_answer, Params: question_text, Returns: "This is your question: **question_text**".

Front-end – can be developed as a web-page with chat window OR messenger's (Telegram, Whatsapp) API can be used.

Final goal – connect your Front-end with Back-end.

The complete task will be published in 1-2 days.

1. SPARQL;
2. Work with Natural Language (NER);
3. **Question classification**;
4. Back-end and Front-end;
5. Simple QA system;
6. Tests for QA system;
7. Docker;
8. Qanary Framework;
9. ...