CAS in NLP

- Module 3 – Day 1 - NLP

- Ahmad Alhineidi

Agenda – day 1

- 8:15 – 9:30: Recap (NLP tasks & overview)
- 9:30 – 10:00: NLP approaches & open source modules
- 10:00 – 10:30: Coffee break
- 10:30 – 12:30: Introduction to Neural Networks (Mykhailo)
- 12:30 - 17:00 Lunch bag and free time, work, sleep, swim, contemplate
- 17:00 – 19:00: Model Context Protocol (MCP)

CAS in NLP

Example scripts are on this github repo

Natural Language?

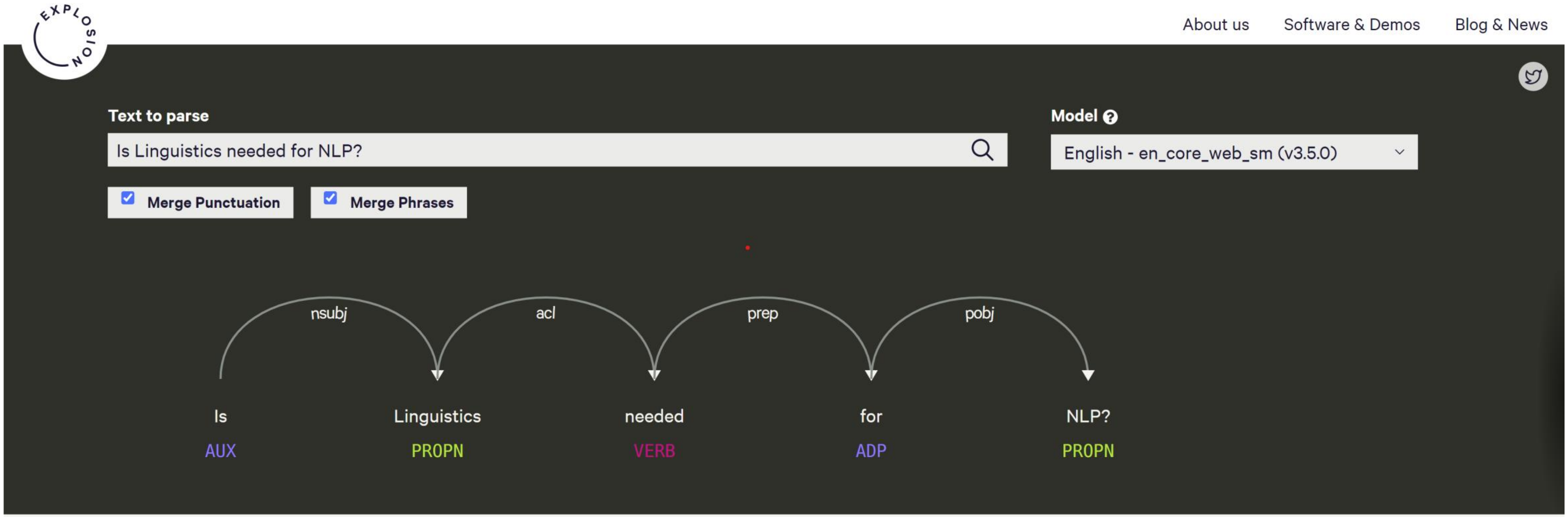
- “…network of constructions..”
- -“C is a construction iff C is a form- meaning pair <F, S> such that some aspects of F or some aspects of S is not strictly predictable from C’s component parts or from other previously established constructions.”

Goldberg, A. E. (1995). *Constructions: A construction grammar approach to argument structure*. University of Chicago Press.

[Adele Goldberg on Linguistics and Grammar (Youtube)]

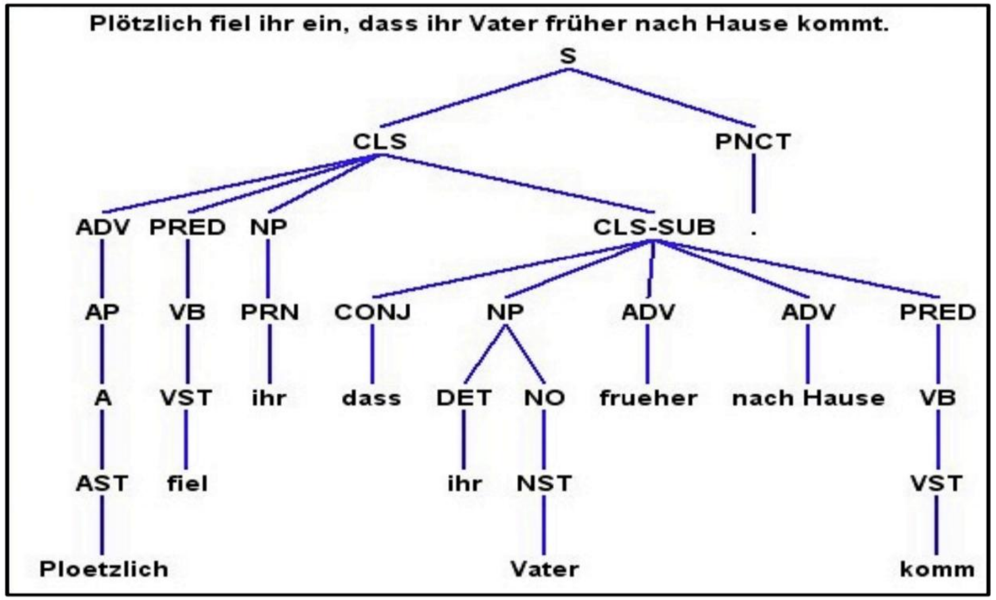
Linguistics for NLP?

- How much linguistic knowledge needed for NLP?
- Will a POS tagged corpus perform better as training data for machine learning algorithm?



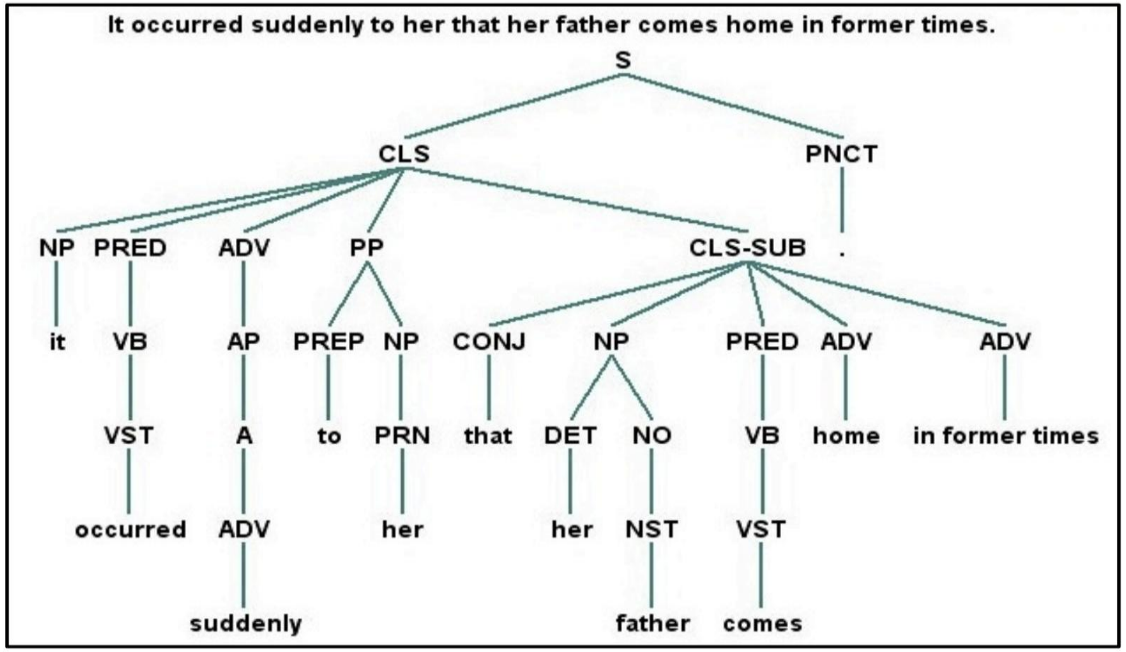
Linguistics for NLP?

Old machine translation system relied on linguistic knowledge



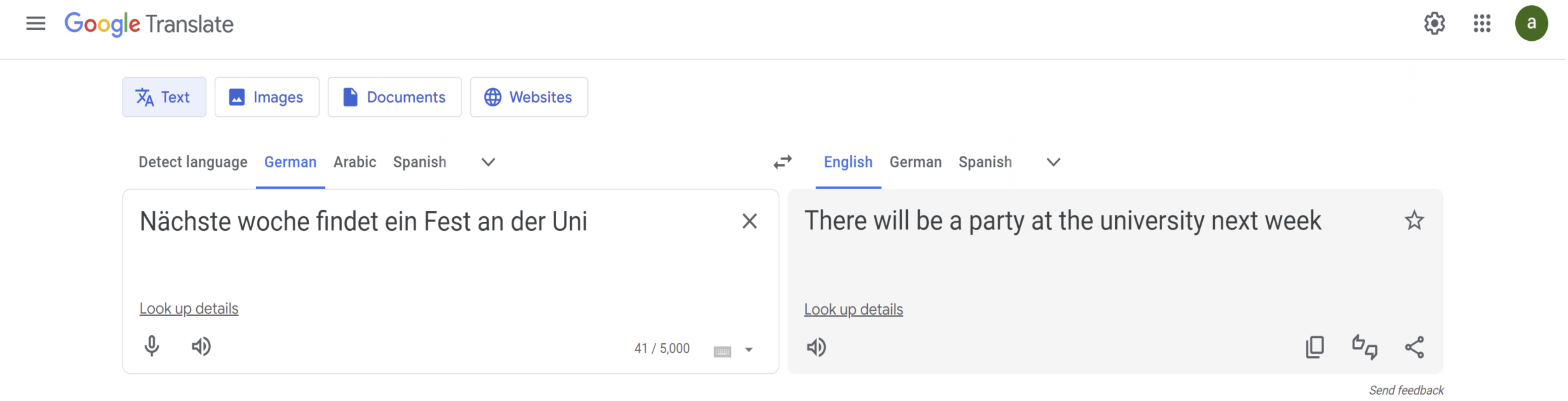
Linguistics for NLP?

Old machine translation system relied on linguistic knowledge



Linguistics for NLP?

New machine translation system predict the next word or sequence



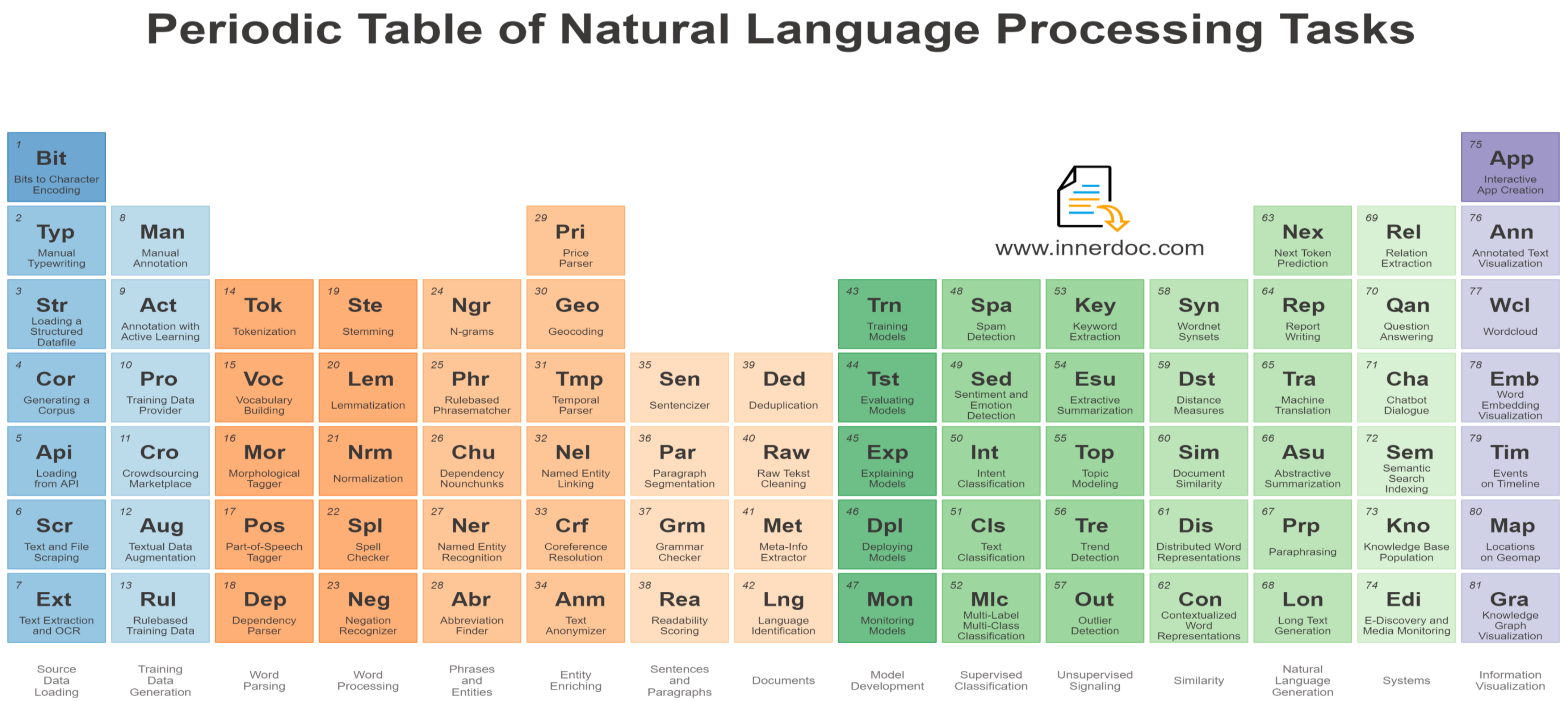
Linguistics for NLP?

- “Anytime a linguist leaves the group, the recognition rate goes up”

(1988), Fred Jelinek, pioneer in Statistical methods for Speech Recognition

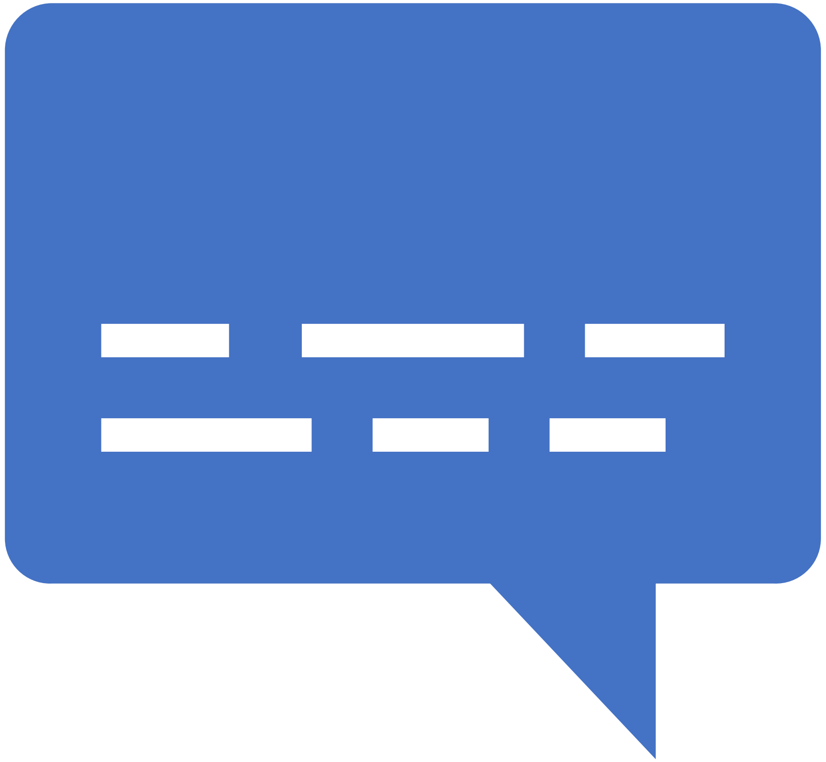
NLP tasks

- Source: Innerdoc link



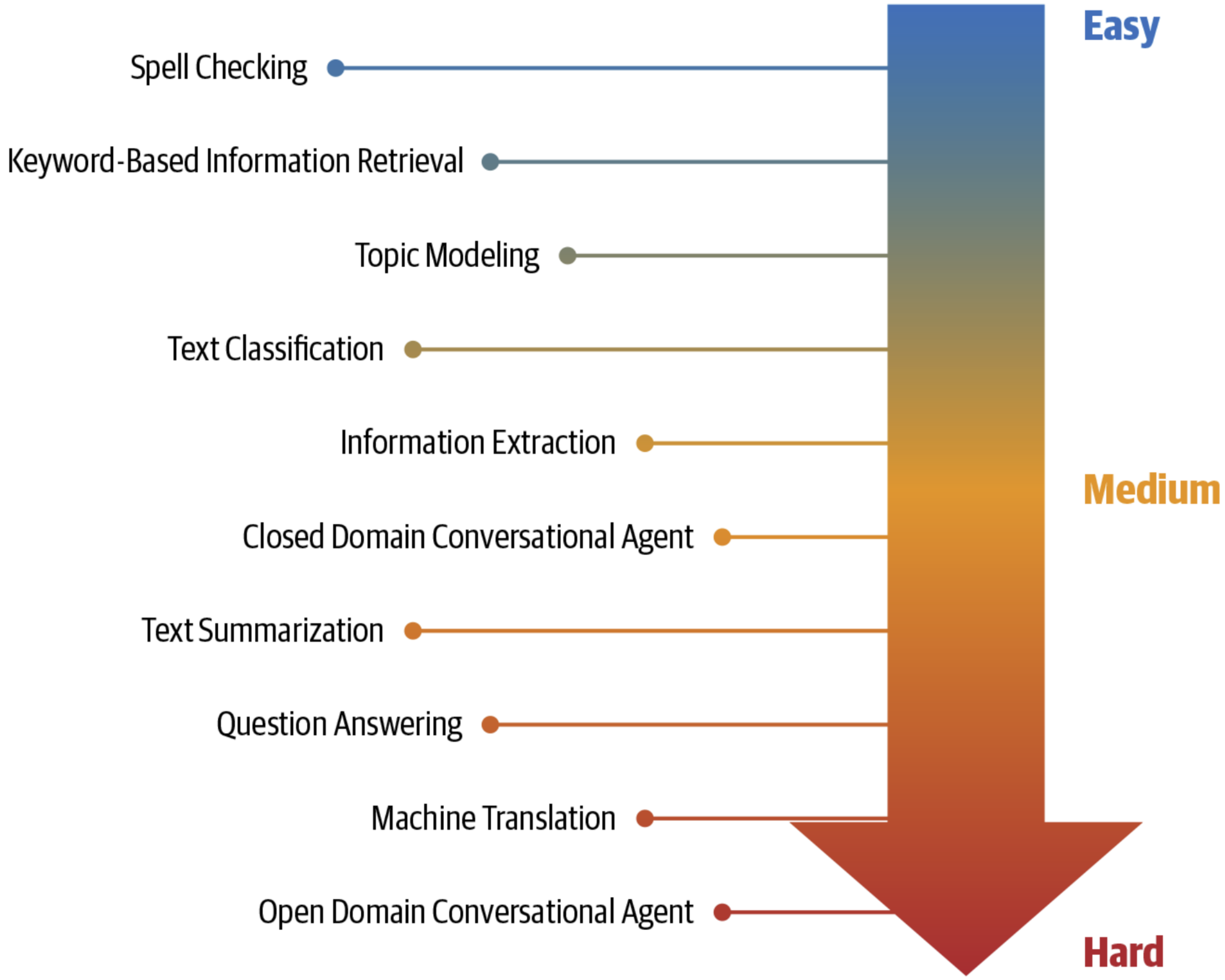
Common NLP tasks and applications

- Text classification (Sentiment analysis, spam detection, topic labeling)
- Named Entity Recognition (NER) (Information extraction, content recommendation)
- Machine Translation (Content Localization, real-time translation)
- Text Summarization (News Aggregation, Research)
- Question Answering (Customer Support)
- Speech Recognition (Voice Assistants)
- Text Generation (Content Creation, chatbots)
- Topic modeling, clustering (Information extraction)



Common NLP tasks

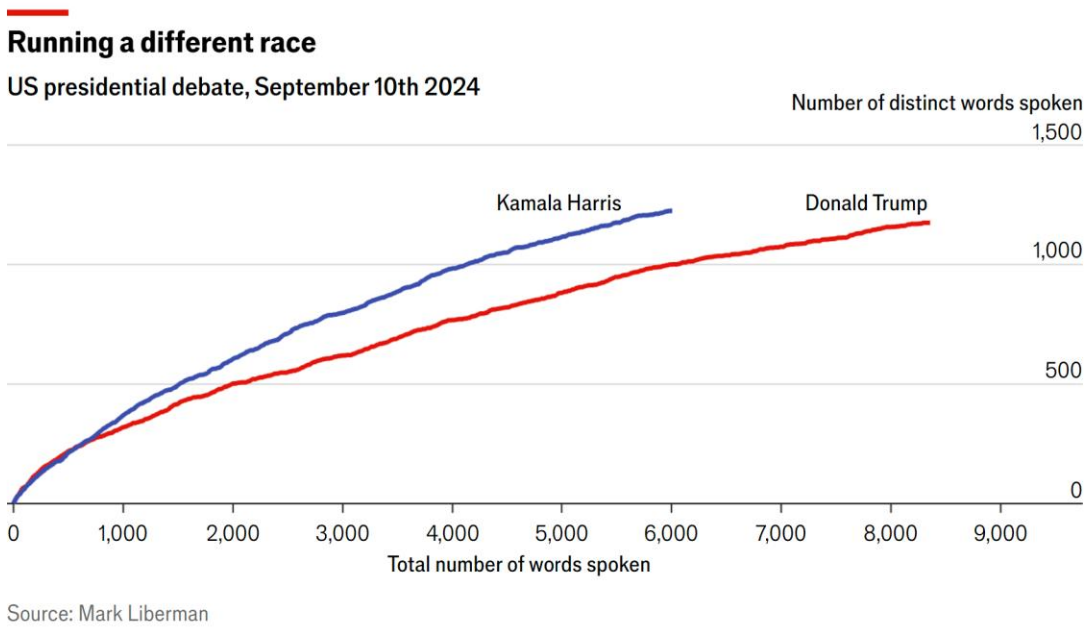
Source: [link]



Source: Vajjala et al. 2020

- Given the debate transcript, how to solve this task?
- Can you produce similar results on previous debates? Link
- Post results here if you do so

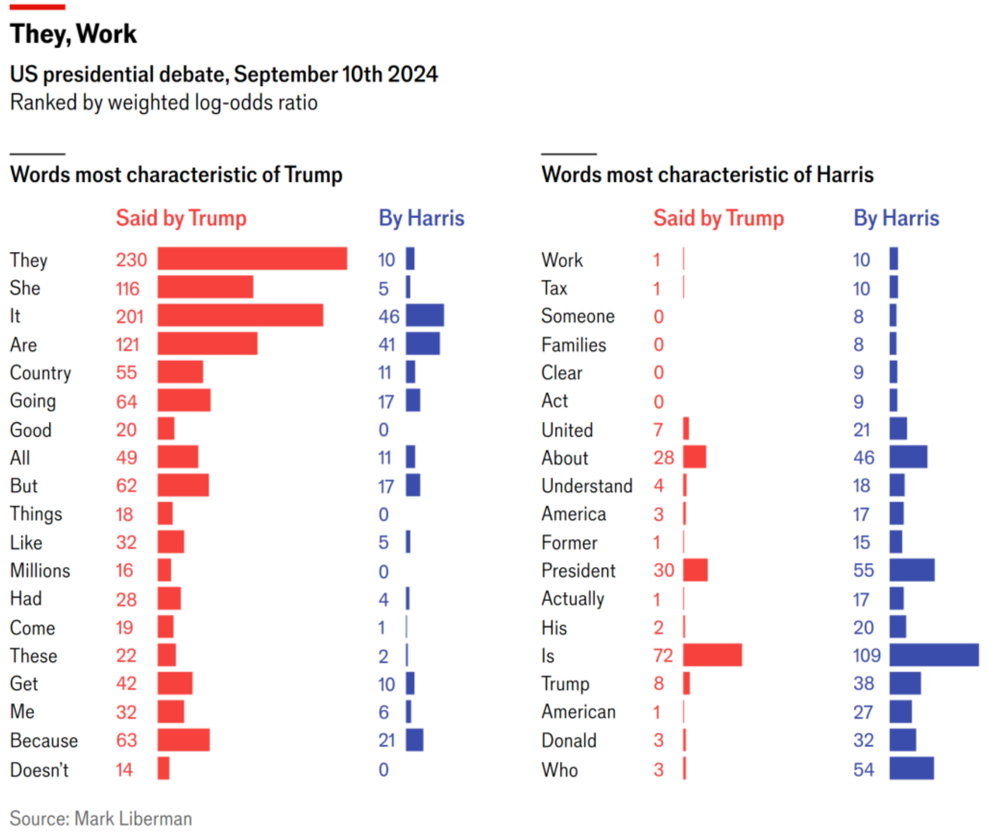
NLP tasks



Source: Link

- Given the debate transcript, how to solve this task?
- Can you produce similar results on previous debates? Link
- Post results here if you do so

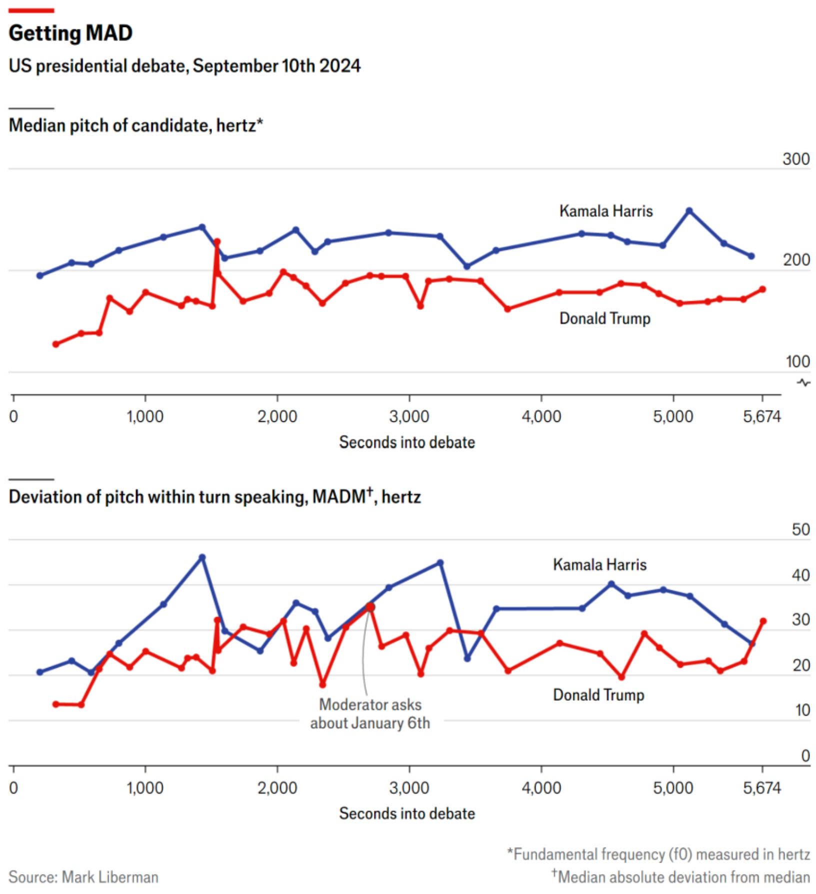
NLP tasks



Source: Link

- Given the audio file of the debate, can you produce similar results?
- Can you produce similar results on previous debates? Link
- Post results here if you do so

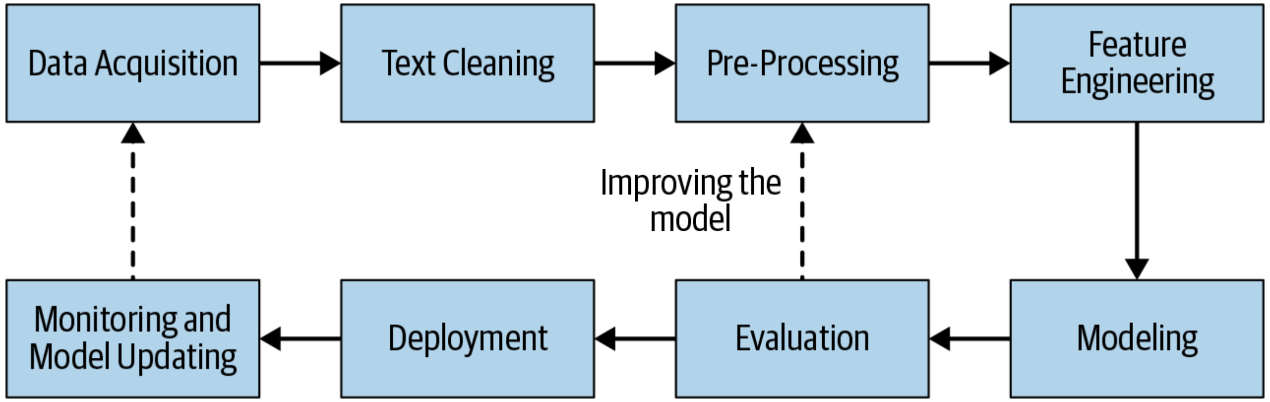
NLP tasks



Source: Link

NLP process

Generic NLP Pipeline

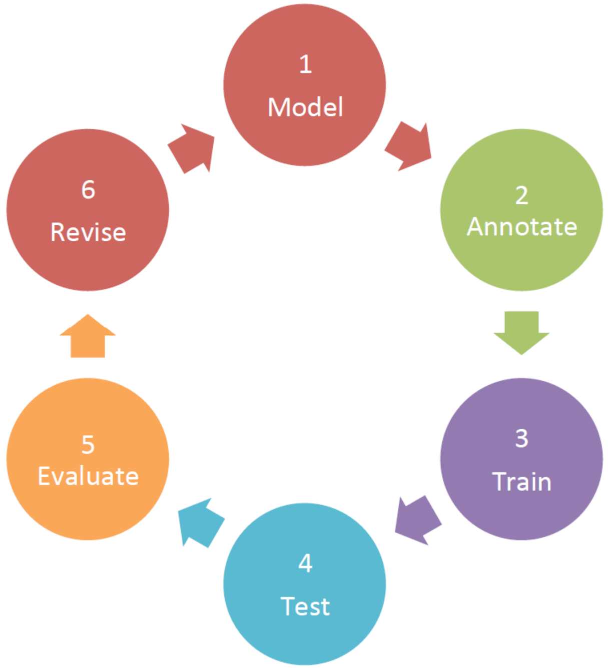


Source: Vajjala et al. 2020

MATTER Model of Annotation for Machine Learning

- Annotation cycle [Pustejovsky and Stubbs 2013]:

  1. model the phenomenon of interest and create guidelines for annotation.
  2. annotate real data
  3. train statistical model
  4. apply model to test data
  5. evaluate results
  6. revise policies, data, and/or learning procedures



Source: Pustejovsky, J and A. Stubbs (2013). Natural language annotation for machine learning. O'Reilly Media, Sebastopol, CA.

NLP progress?

The model with the best performance wins. Is that the only metric?

Check https://nlpprogress.com/

Commercial NLP APIs

- Explore one the demos of commercial tools for NLP (text mining):

  2. Google Cloud Natural Language
  4. Pikes
  5. TextRazor
  6. text2data
  7. Dandelion
  8. Gate Cloud

- Give a feedback of what you like and dislike
- Compare results by prompting chatgpt/gemini
- Use the link to add screenshot, comments of your findings link

NLP approaches

RULE AND LOGICAL BASED

STATISTICAL (CLASSICAL ML) MODELS

NEURAL NETWORKS (DEEP LEARNING)

NLP approaches

- Source: Schmidt, Thomas, et al. "Sentiment analysis on twitter for the major German parties during the 2021 German federal election." *Proceedings of the 18th Conference on Natural Language Processing (KONVENS 2022)*. 2022.

	SVM	**NB**	**GerVADER**	**BERT-1**	**BERT-2**	**BERT-3**
Accuracy	57.6	65.0	52.0	85.8	81.5	**93.3**
F1 Macro	54.5	65.3	52.0	82.1	73.8	**93.4**
F1 Weighted	55.9	65.1	54.0	85.9	81.5	**93.3**

Table 4: Results of the evaluation of the different sentiment analysis approaches. Best results per metric are marked in bold.

NLP approaches

Experiment result	Accuracy in percentage			
	Feature Extraction Techniques			
Algorithms	BOW	TF-IDF	Pre-trained Word2vec	Embedd-ing Layer
SVM	0.78	0.80	0.82	-
NB	0.80	0.80	0.74	-
RF	0.79	0.79	0.81	-
XGBoost	0.80	0.77	0.81	-
CNN	-	-	0.81	0.82
BI-LSTM	-	-	**0.84**	0.81

Table 5: Eight classes experiment result with classical, en-semble, Deep ML classifier

- Source: Ababu, Teshome Mulugeta, and Michael Melese Woldeyohannis. "Afaan Oromo hate speech detection and classification on social media." Proceedings of the thirteenth language resources and evaluation conference. 2022.

Rule-based NLP

- Example one: Tokenizer
- Rule 1: Replace every punctuation with “white space + punctuation”,
  “I like apples.” -> “I like apples .”
- Rule 2: Replace every white spaces with newline
- Python implementation “tokenizer.ipynb”

Rule-based NLP

- Example two: Language identifier
- Step 1: given corpora in different languages, extract most 100 frequent bigrams or trigrams [fleets -> [“fl”, “le”, “et”, “ts”] or [“fle”, “lee”, “ets”]
- Step 2: convert the input text into bigrams or trigrams
- Step 3: calculate a score between the bigrams or trigrams of the input text with the most frequent bigrams and trigrams from each language
- Calculate a score of the number of matches and choose the language with the highest score
- Python implementation “lan_identifier.ipynb

Rule-based NLP

- Example three: Sentiment analysis
- VADER (Valence Aware Dictionary and sEntiment Reasoner)
- Nothing to do with Star Wars Vader
- Lexicon and rule-based sentiment analysis tool
- Built for sentiment analysis in social media
- Lexicon: (large vocabulary [pos, neg], valence score for each word)
- Rules: (punctuation, capitalization, adverbs usage, conjunction and negation, etc)
- Doesn’t generalize well, fail with mixed sentiment
- Example code implementation on github
- NLTK code implementation on NLTK, NLTK Tutorial on Sentiment Analysis
- Code example of using NLTK vader for sentiment analysis

Rule-based NLP

- Discuss in group of 2 - 3 the following (5-10min):

1- What are the pros and cons of using such approaches for example 1 & 2?

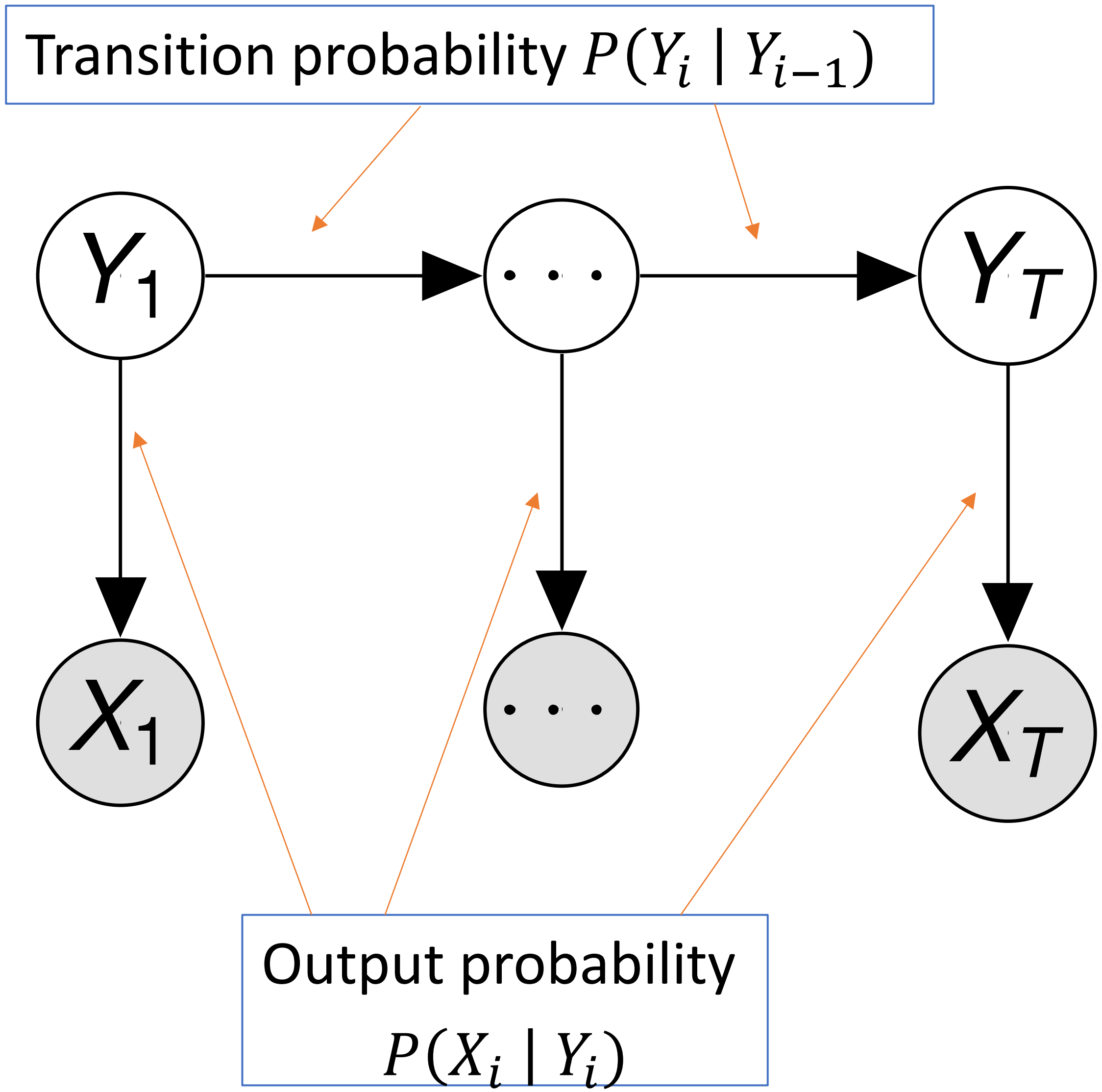
2- Find examples where the tokenizer or the language identifier would fail

Statistical NLP (HMM)

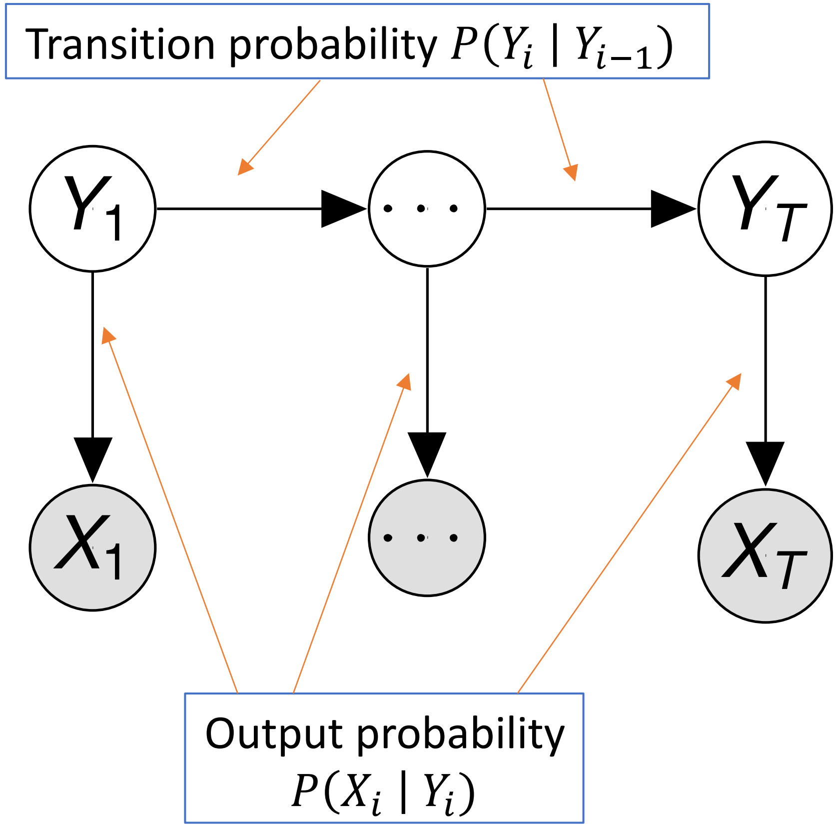
- Relies heavily on probability theory
- Example: Hidden Markov Model (HMM) for POS tagging
- Given a PoS-tagged training corpus, HMM model calculates the joint probability distribution P(X, Y): What is the probability of observing a sequence of words x with PoS-tag labels y?
- From this joint probability P(X,Y), we can then infer the conditional probability P(Y | X) that given a certain sequence of words x the correct PoS-tag labels are y by applying Bayes rule.
- After having probability distribution, we chooses the best label sequence with argmax.

Statistical NLP (HMM)

- Observed events $\boldsymbol{x\_1}, \ldots \boldsymbol{x\_T}$: words/tokens that we can see in the input
- Hidden events $\boldsymbol{y\_1}, \ldots \boldsymbol{y\_T}$: part-of-speech tags that we think of as causal factors for the observed events.
- Assumption 1: Each token only depends on the current part-of-speech
- Assumption 2: Each part-of-speech depends only on the immediately preceding part-of-speech



Statistical NLP (HMM)



Example: Transition probability

	NNP	MD	VB	JJ	NN	RB	DT
<s>	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

Figure 8.7 The A transition probabilities $P(t_i|t_{i-1})$ computed from the WSJ corpus without smoothing. Rows are labeled with the conditioning event; thus $P(VB|MD)$ is 0.7968.

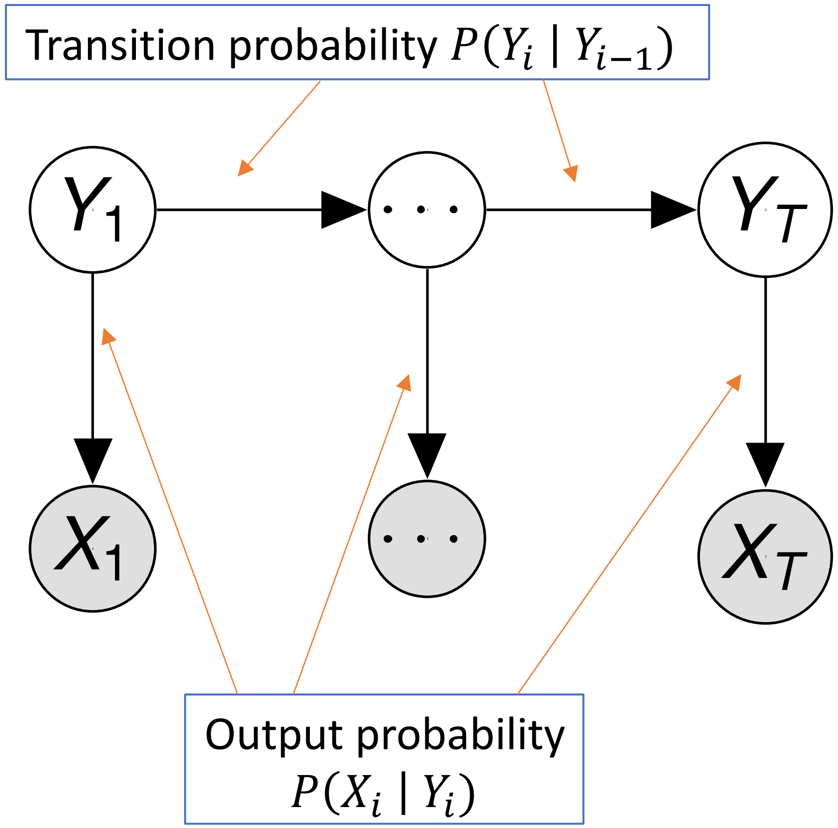
Example: Output probability

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

Figure 8.8 Observation likelihoods B computed from the WSJ corpus without smoothing, simplified slightly.

Source: Jurafsky & Martin, Speech and Language Processing, 2019

Statistical NLP (HMM)



Joint probability that a certain sequence of words $x_1, \ldots, x_T$ with PoS-tags $y_1, \ldots, y_T$ occurs

$$P(Y_1, \ldots, Y_T, X_1, \ldots, X_T) = P(Y_1)P(X_1|Y_1) \prod_{t=2}^{T} P(Y_t|Y_{t-1})P(X_t|Y_t)$$

$P(Y_1)$: Probabilities for the first PoS-tag of a sequence

$P(Y_t|Y_{t-1})$: Transition probabilities: conditional probability of a PoS-tag given the immediately preceding PoS-tag

$P(X_t|Y_t)$: Output probabilities conditional on the PoS-tag (including $P(X_1|Y_1)$)

Statistical NLP (HMM)

- After training HMM for POS tagging, we can predict POS tags for a sequence of tokens
- We use argmax function

Example:
Given the token sequence `The man tries` find the most likely sequence of PoS-tags:

$$\underset{\boldsymbol{y}\,\in\{(DT,NN,VBZ),(NN,VBZ,DT),(DT,NS,NS),\ldots\}}{\operatorname{argmax}} P(Y = \boldsymbol{y} \mid X = (\texttt{The, man, tries})) = (DT, NN, VBZ)$$

NLTK (Natural language toolkit)

- 2001 – present
- University of Pennsylvania
- Statistical NLP in Python
- classification, tokenization, stemming, tagging, parsing, others
- https://www.nltk.org/

NLTK (Natural language toolkit)

- Pre-trained tokenizers (Twitter-aware tokenizer, statistical, other)

```
>>> import nltk
>>> nltk.download("punkt")
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\ahmad\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
>>> text = "Programming with python is fun."
>>> from nltk.tokenize import word_tokenize
>>> word_tokenize(text)
['Programming', 'with', 'python', 'is', 'fun', '.']
>>> type(word_tokenize(text))
<class 'list'>
```

NLTK (Natural language toolkit)

- Sentence tokenizer

```
>>> import pprint
>>> from nltk.tokenize import sent_tokenize
>>> text = "Programming is fun in Python. NLP stands for natural language processing. NLTK is a great module for NLP."
>>> pprint.pprint(sent_tokenize(text))
['Programming is fun in Python.',
 'NLP stands for natural language processing.',
 'NLTK is a great module for NLP.']
>>>
```

NLTK (Natural language toolkit)

- Wordnet lemmatizer

```
>>> import nltk
>>> nltk.download("wordnet")
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\ahmad\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
True
>>> from nltk.stem import WordNetLemmatizer
>>> lemmatizer = WordNetLemmatizer()
>>> lemmatizer.lemmatize("corpora")
'corpus'
>>> lemmatizer.lemmatize("better", pos='a')
'good'
>>> lemmatizer.lemmatize("stones")
'stone'
```

NLTK (Natural language toolkit)

- Many corpora for different purposes (Stopwords removal, training sets)

```
>>> from nltk.corpus import stopwords
>>> stopwords = stopwords.words("english")
>>> stopwords[:5]
['i', 'me', 'my', 'myself', 'we']
>>> text = "This is an example for stopwords removal by using NLTK"
>>> text = text.split() # return a list with each word as an element (split the text based on whitespaces)
>>> text = [word.lower() for word in text if word not in stopwords]
>>> text
['this', 'example', 'stopwords', 'removal', 'using', 'nltk']
>>> stopwords.extend(["this"])
>>> text = [word.lower() for word in text if word not in stopwords]
>>> text
['example', 'stopwords', 'removal', 'using', 'nltk']
```

NLTK (Natural language toolkit)

POS tagging

```
>>> from nltk.tag import pos_tag
>>> from nltk.tokenize import word_tokenize
>>> pos_tag(word_tokenize("John's big idea isn't all that bad."))
[('John', 'NNP'), ("'s", 'POS'), ('big', 'JJ'), ('idea', 'NN'), ('is', 'VBZ'),
("n't", 'RB'), ('all', 'PDT'), ('that', 'DT'), ('bad', 'JJ'), ('.', '.')]
>>> pos_tag(word_tokenize("John's big idea isn't all that bad."), tagset='universal')
[('John', 'NOUN'), ("'s", 'PRT'), ('big', 'ADJ'), ('idea', 'NOUN'), ('is', 'VERB'),
("n't", 'ADV'), ('all', 'DET'), ('that', 'DET'), ('bad', 'ADJ'), ('.', '.')]
```

Source: NLTK documentation

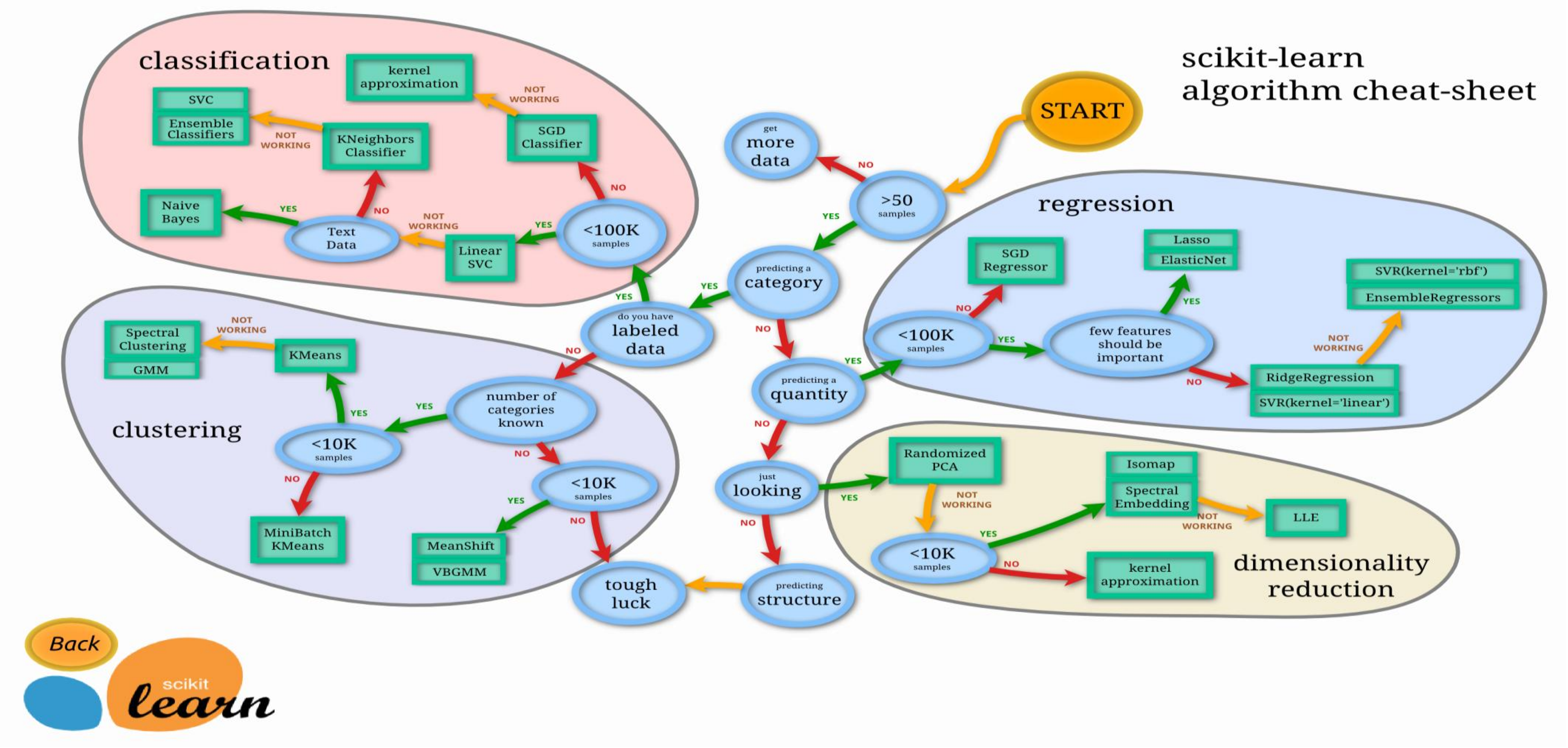
NLTK (Natural language toolkit)

- Hidden Markov Model in NLTK link to code
- HMM Documentation
- Code example of using NLTK HMM to train POS tagger
- Code in hmm_nltk.py or hmm_nltk.ipynb

scikit-learn (sklearn)

- Machine learning open-source library
- Many implementation for NLP algorithms
- Classification, regression, clustering, word embedding
- SVM, Random forests, k-means, others
- Built for python, works well with numpy, scipy
- Great documentation: User guide
- Real-life examples: Examples

scikit-learn (sklearn)



Source: https://scikit-learn.org/stable/_static/ml_map.png

scikit-learn (sklearn)

- Pipeline in sklearn (from sklearn.pipeline import Pipeline)
- simplifies the process of building and evaluating machine learning models
- improved code readability (execute several steps in one)
- Code example (sklearn_randomforest.py, sklearn_randomforest.ipynb)

Textblob for text classification

- Using the dataset IBDM reviews Data link
- The data is available in json format on github (train_data.json, test_data.json) link to data
- Use textblob to train a model on the data. Try different number of instances to train the model and compare the accuracy results
- Textblob documentation Documentation
- Link to building a text classifier
- Run these two commands before you start: (“pip install -U textblob”, “python -m textblob.download_corpora lite”)
- Work in groups of 2-3
- Post your results (accuracy, etc) and observation to this link