Code Structure and Usage

NP_Chunking ~/Desktop/NP_Chunking		
	${\mathbb V}$	1_SynTagRus_Projective
		projectivized-syntagrus_conll2007.txt
	$\overline{\mathbb{W}}$	2_SynTagRus_NP_Chunks
		test.txt
		train.txt
	${\mathbb W}$	<u>□</u> Models
		BIGRAM.pkl
		CRF.pkl
		SKLEARN_MAXENT.pkl
		SVM.pkl
		TRIGRAM.pkl
		UNIGRAM.pkl
	$\overline{\mathbb{V}}$	□ Pipeline
		<pre>initpy</pre>
		🔒 Chunker.py
		🔒 Converter.py
		🔒 main.py
		Feature_extraction.py
		NP Chunker Manual.pdf
		pickler.py
		SynTagRus_NP_Chunk_Corpus_Reader.py

- > 1 SynTagRus Projective contains the text file with the dependency trees of SynTagRus
- > 2 SynTagRus NP Chunks contains the test and train data for the Chunker
- ➤ Models contains the models that will be pickled during Chunker training
- ➤ The *Pipeline* package contains:
 - Converter.py: the code for converting the dependency trees from 1_SynTagRus_Projective into the Conll-style train and test data, storing it in 2_SynTagRus_NP_Chunks
 - *Chunker.py*: the Chunker class with the methods to extract the train and test data and apply the different models to it. A Chunker object must be instantiated inside *main.py* with one of the following parameters:

```
{'UNIGRAM', 'BIGRAM', 'TRIGRAM', 'SKLEARN_MAXENT', 'CRF', 'SVM'} (See commented code in main.py)
```

- o *main.py*: the code used to train and evaluate models
- > feature_extraction.py contains the code to extract the different features from data. Edit features inside the features method of this script
- > SynTagRus_NP_Chunk_Corpus_Reader.py: adapted from nltk conll corpus reader to read converted train and test data
- > pickler.py is basic code to pickle and load models

Required Installations

- pycrfsuite (<u>https://python-crfsuite.readthedocs.io/en/latest/</u>)
- sklearn (<u>http://scikit-learn.org/stable/</u>)