Assignment 1

In the first part of the assignment we used the NLTK package available in python.

For every test, we used several baseline taggers in order to see if there were some difference between one baseline tagger and another.We used the taggers ‘Default’, ‘Unigram’ and ‘TnT’.

We runned several tests with different numbers of max_rules and templates in order to see if there were some relevant improvements when going from one method to another.

The results are reported in the following tables:

TEXTEN2.ptg

Max_Rules = 100, Template = fntbl37

Split	Default	Unigram	TnT
Split 0	0.6534299	0.8593591	0.8667118
Split 1	0.6515234	0.8431203	0.8479683
Split 2	0.6711736	0.8624983	0.8655203
Split 3	0.6491694	0.8475498	0.8501796
Split 4	0.6598238	0.8570051	0.8597926
Mean	0.6570240	0.8539065	0.8580345
Variance	6.2583883e-05	5.3979986e-05	5.9490577e-05

Max_Rules = 180, Template = fntbl37

Split	Default	Unigram	TnT
Split 0	0.6464422	0.8586551	0.8650170
Split 1	0.6453202	0.8412437	0.8464305
Split 2	0.6657809	0.8614823	0.8639572
Split 3	0.6444826	0.8461698	0.8497370
Split 4	0.6535195	0.8552857	0.8585161
Mean	0.6511091	0.8525673	0.8567316
Variance	6.4054555e-05	5.8615830e-05	5.5816065e-05

Max_Rules = 250, Template = brill24

Split	Default	Unigram	TnT
Split 0	0.6698563	0.8618621	0.8682241
Split 1	0.6678134	0.8466651	0.8505486
Split 2	0.6882115	0.8639312	0.8667187
Split 3	0.6723689	0.8489038	0.8514555
Split 4	0.6735007	0.8598186	0.8625801
Mean	0.6743502	0.8562362	0.8599054
Variance	5.1941467e-05	4.9813781e-05	5.6345379e-05

TEXTCZ2.ptg

Max_Rules = 100, Template = fntbl37

Split	Default	Unigram	TnT
Split 0	0.3748339	0.6121207	0.6173282
Split 1	0.4324618	0.6653913	0.6726978
Split 2	0.4085242	0.6344684	0.6395607
Split 3	0.4126288	0.6164332	0.6247556
Split 4	0.3892601	0.5876541	0.5912992
Mean	0.4035418	0.6232136	0.6291283
Variance	0.0003943	0.0006678	0.0007193

Max_Rules = 300, Template = fntbl37

Split	Default	Unigram	TnT
Split 0	0.39677985	0.6175939	0.62160582
Split 1	0.45066157	0.6700674	0.67599235
Split 2	0.43106832	0.6400912	0.64521006
Split 3	0.42594452	0.6228005	0.62882431
Split 4	0.40864788	0.5941255	0.59718957
Mean	0.42262042	0.6289357	0.63376442
Variance	0.00034633	0.0006388	0.00068483

Max_Rules = 500, Template = brill24

Split	Default	Unigram	TnT
Split 0	0.42350816	0.62505978	0.62792922
Split 1	0.47125246	0.67721452	0.68048249
Split 2	0.45589327	0.64711967	0.65138977
Split 3	0.44832232	0.63138705	0.63450462
Split 4	0.43461264	0.60173275	0.60558916
Mean	0.44671777	0.63650275	0.63997905
Variance	0.00027478	0.00062724	0.00062571

Conclusions:

From the results reported here, there are several observations that can be done:

1) **Tagger:**
   The best baseline tagger to use is the UnigramTagger.
   TnT tagger reported better performance in every case, however the improvement was relatively small (around 0.4% on average).Being the TnT more complex, the training time needed for this tagger is higher and, in addition, we have higher change of overfitting the data.

2) **Max rules:**
   a. For the English text, we can see that by increasing the number of rules the performance achieved get slightly lower. This is probably due to the fact that more

than 100 rules in English lead to a higher chance of overfitting.
100 rules is the best choice from our tests

b. For the Czech text, we can see a slight improvement (around 0.5% on average) by incrementing the number of rules used. However, when doing so, the training time increases a lot (from 1.30 hours for every split to 4.30 hours on my hardware). Therefore, the choice on whether to increase the number of rules or not, strongly relies on the hardware used in the training phase

3) **Templates:**
We can see that by changing the template from fntbl37 to brill24 and increasing the number of rules, both for the English and Czech file, we have an increase in the performance of around 0.7%. The training times are similar to the 2nd test case but the performance are better.
brill24 seems to be better a better choice than fntbl37. Maybe having many possible templates can increase the chance of overfitting the model.

Further tests could be done on a proper hardware, that will lead to shorter training time for the models (with my hardware the training for the simplest model lasted 6 hours) in order to pick the best set of templates/max rules/baseline taggers by the usage also of a validation set.