Als erstes musste man das Europarl-Korpus für das Training aufbereiten (Preprocessing). Als erstes haben wir alle Sonderzeichen im Text normalisiert (z.B. Anführungs- und Schlusszeichen vereinheitlicht). Wir brauchten kein Cleaning (bspw. von Smileys,…), da es sich um das Europarl-Korpus handelt. Als zweites haben wir den Text tokenisiert, damit jedes Token einzeln steht. Das Datenset wurde schon durchmischt, daher konnten wir direkt das Truecasing und anschliessend das BPE machen. Beim Truecasing haben wir den Text „getruecased“, d.h. wir haben ein gelerntes Modell benutzt, um zu entscheiden, ob wir bspw. am Satzanfang die Grossschreibung der Tokens beibehalten wollen oder nicht. Wir haben uns für die Kleinschreibung der Tokens am Satzanfang entschieden. Anschliessend haben wir das BPE-Modell trainiert und BPE auf den Text angewendet. Beim BPE haben wir die häufigsten zusammen erscheinende Sequenzen von Symbolen zusammengefügt. Wir hatten bei dieser Übung etwas Schwierigkeiten, daher haben wir das Postprocessing ausgelassen. Die Schritte beim Post-Processing wären jedoch: Reverse BPE, Detruecasing (d.h. die Wörter wieder in ihre ursprüngliche Form, also bspw. auf Deutsch Nomen und auf Englisch Nationalitäten und Länder den ersten Buchstaben des jeweiligen Wortes gross schreiben) und schliesslich die Detokenisierung (z.B. würde der „Punkt“ auf Englisch oder Deutsch wieder am letzten „Wort-Token“ angehängt, sowie das „Komma“ oder der „Doppelpunkt“ am vorherigen Token,…). Wir haben die PostProcessing-Skripte von Ondřej Dušek(https://github/ufal/mtmonkey) genommen und etwas im Codes abgeändert. Diese Schritte sind uns theoretisch klar, aber noch nicht in der Praxis. Ausserdem verstehen wir den Reverse BPE nicht ganz, also den Grund weshalb man diesen braucht?

Der erste übersetzte Text sah katastrophal aus. Bei diesem haben wir die Vokabulargrösse auf ihre Ursprungsform gelassen, also 50000. Dabei haben wir das Datenset für 6 Epochen trainiert. Der zuletzt übersetzte sieht am besten aus. Bei diesem hatten wir eine Vokabulargrösse von 90‘000 und das Datenset auf 6 Epochen trainiert. Wir konnten feststellen, dass ein grösseres Vokabular bessere Resultate liefert. Wir haben uns gefragt, ob je mehr Vokabular desto besser oder, ob es irgendwann ein „Kipppunkt gibt d.h., ob irgendwann zu viel Vokabular zu einer Verschlechterung führen könnte? Je mehr Epochen desto eher besteht die Gefahr, des Overfittings (d.h. das System merkt sich die „Lösung“ bzw. es lernt einfach auswendig. Man kann dies beispielsweise so verhindern, dass man in jedem Satz ein Wort als „<unk>“ unknown setzt, damit sich das System nicht daran gewöhnen kann) und es dauert zeitlich viel länger, deswegen haben wir uns für 6 Epochen entschieden. Mit der Option --sample after epoch kann man das Overfitting verhindern, da sich sobald die Werte verschlechtern die Gefahr des Overfittings erhöht. Wir haben auch versucht, mehr LSTM Layers einzubauen, damit wir das NMT-System tiefer machen können, aber wir hatten keinen Erfolg bei der Codeänderung und haben ständig Fehlermeldungen erhalten.