# Bericht MT Übung 4 - RNNs

## Welches Datenset habt ihr verwendet?

Wir haben uns für die spanische Originalversion von Don Quijote de la Mancha entschieden. Dies erschien uns interessant, da viele altspanische Formulierungen darin vorkommen. Zudem bietet der Text eine genügend grosse Datenmenge und ist in sich sehr homogen (soll heissen, dass wir dadurch eventuell später einen qualitativ hochwertigeren Text produzieren können als mit einem Datenset, das aus verschiedenen Texten von verschiedenen Autoren und aus verschiedenen Epochen stammt).

## Wie gestaltete sich das Preprocessing?

Der Text wurde von der gutenberg.org-Seite heruntergeladen und stand uns als .txt-Dokument zur Verfügung. Manuell haben wir dann allen Text, der nicht Spanisch war (also die gesamte Präambel und die Schlussdeklaration) aus dem Dokument gelöscht. Danach mussten wir den Text tokenisieren, da *romanesco* verlangt, dass das Datenset aus einem Satz pro Zeile besteht und dieser Satz in sich jeweils durch Leerzeichen (inkl. Satzzeichen) getrennt ist. Dazu benutzten wir den Satz- und den Wort-Tokenizer von NLTK. Das Preprocessing-Programm liegt im Repository unter *preprocess.py* bereit.

## Adaptionen

Wir entschieden uns dazu, ein Hidden Layer der Grösse 1000 einzufügen (siehe *compgraph.py*). Dies verbesserte die Performance um ca. 6% von einer Perplexität von 84.53 auf 79.79. Zuerst hatten wir Probleme damit, das Hidden Layer korrekt zu implementieren. Zudem wurde erst übersehen, dass die Grösse des Hidden Layers im separaten *const.py*-File definiert werden sollte.

## Hyperparameter

Learning Rate: Zusätzlich zum Hidden Layer wurde die ursprüngliche Learning Rate einmal um 10 erhöht und einmal um 100 erhöht. Dabei erzielten wir das beste Resultat mit einer Learning Rate von 0.001, was zu einer Endperplexität von 29.84 führte.

Neuronen des Hidden Layer: Wir haben auch mit der Anzahl Neuronen im Hidden Layer gespielt und diese versuchshalber auf 1000 und 1500 gesetzt. Dabei wurde allerdings schnell klar, dass 1500 Layers zu einem Overfitting-Problem führen und man somit das Training um sehr viele Epochen reduzieren müsste, was uns nicht sehr zielführend erschien.

Epochen: Beim Aufruf der Trainings-Funktion haben wir die Anzahl der Epochen reduziert, um einem Overfitting entgegenzuwirken, da wir die Learning Rate auf 0.001 gehoben haben. Wir haben uns für 6 Epochen entschieden, da so die Perplexität des Scorings immer noch tief ausfällt. Dies haben wir dadurch bestimmt, dass mit 10 Epochen der Unterschied zwischen der Perplexität der 10. Epoche und des Scorings sehr gross war. Der Unterschied zwischen der 6. Epoche und dem Scoring war der geringste.

## Perplexität

Training ohne Adaptionen: 84.53

Mit 2 RNNs, Learning Rate 0.0001, 10 Epochen: 109.74

Mit Hidden Layer (1000 Neuronen), Learning rate 0.0001, 10 Epochen: 79.79

Mit Hidden Layer (1000 Neuronen), Learning Rate 0.001, 10 Epochen: 29.84

Mit Hidden Layer (1500 Neuronen), Learning Rate 0.001, 10 Epochen: 5054.11 -> Overfitting!

**Mit Hidden Layer (1000 Neuronen), Learning Rate 0.001, 6 Epochen: 26.14**