Report Assignment 4

Korpus

Für diese Übung wurde das *United Nation General Debate* Korpus verwendet (Baturo et al. 2017), das ich ebenfalls in meiner Masterarbeit verwenden werde. Der Korpus ist von mittlerer Grösse, wie an den deskriptiven Kennzahlen aus Tabelle 1 zu sehen ist. Der Korpus wurde mithilfe von spaCy segmentiert und tokenisiert. Während in der Basisversion *v1_basic* die originale Gross- und Kleinschreibung beibehalten wurde, ist in Version *v2_adv* ein Truecasing für die Wörter am Satzanfang umgesetzt und korrigiert zudem einen Fehler, bei dem durch Tabulatoren tlw. zwei Sätze zusammengehängt sind. Truecasing reduziert die Vokabulargrösse ein wenig und das RNN muss später nicht aufgrund der Satzgrossschreibung verschiedene Sequenzen lernen. Die Grossschreibung könnte beim Sampling einfach regelbasiert wieder in Grossschreibung überführt werden. Der Korpus wurde dann in ein Trainings- und ein Dev-Datensatz aufgeteilt in einem Verhältnis von 90% zu 10%.

version	**preprocessing**	**Descriptive statistics**
v1_basic	Basic preprocessing	Total sentences: 744'612 Total tokens: 25'213'492 Unique tokens: 75'118 Avg. tokens per sentence: 33.86
v2_adv	Truecased, further preprocessing	Total sentences: 819'005 Total tokens: 2'463'1767 Unique tokens: 71'472 Avg. tokens per sentence: 30.08

Tabelle 1: Deskriptive Kennzahlen Korpus

Modell

Auf dem oben beschriebenen Korpus wurden insgesamt drei Sprachmodelle mit verschiedenen Hyperparameter trainiert, deren Spezifikation und Resultate in Tabelle 2 zu sehen sind. Als erstes wurde das originale Vanilla RNN mit allen voreingestellten Hyperparameter trainiert, was eine Perplexity von rund 20 sowohl auf den Trainings- wie auch Dev-Datensatz ergeben hat. Diese geringe Differenz ist erfreulich und weist auf ein geringes Overfitting hin.

Um die Performance zu verbessern wurde dann ein bidirektionales Netz implementiert. Allerdings ist die Implementation für ein Sprachmodell nicht korrekt, wie an der zu geringen Perplexity von knapp über 1 zu sehen ist. Statt nur den linken und rechten Kontext eines zu voraussagenden Wort, wird bei der Backward-Sequence das Zielwort fälschlicherweise ebenfalls schon encodet, was dann zu einer nahezu perfekten Vorhersage führt. Bei der Backward-Sequenz dürfte somit nur immer $t_{\text{Zielwort-1}}$-Zustände als Input-Signal verwendet werden.

Als drittes Modell wurde ein Stacked RNN mit 2 Layern und Dropout trainiert. Beim Stacking werden mehrere Layer hintereinander geschaltet, wobei das Dropout einem Overfitting durch ein zufälliges Deaktivieren von Neuronen vorbeugt. Das Dropout darf nur beim Training aktiv sein, danach sollte die volle Kapazität des RNN genutzt werden. Um die Trainings-Geschwindigkeit zu reduzieren wurde statt einer LSTM-Zelle eine GRU-Zelle verwendet, die mit weniger Parametern auskommt. Zudem wurde die Grösse des Hidden-Layers und der Embeddings entsprechend von Standardwerten für kleinere RNN in der Literatur reduziert, was ebenfalls die Berechnungszeit

Architecture	Data	Perplexity	Hyperparameter
Vanilla RNN (LSTM)	training (v1_basic)	20.29	NUM_STEPS = 35 LEARNING_RATE = 0.0001 HIDDEN_SIZE = 1500 VOCAB = 10000 EPOCH = 10 BATCH_SIZE = 20
	dev (v1_basic)	20.75	
Bidirectional RNN (GRU)	training (v1_basic)	1.08	NUM_STEPS = 35 LEARNING_RATE = 0.0001 HIDDEN_SIZE = 700 VOCAB = 10000 EPOCH = 10 BATCH_SIZE = 20 EMBEDDING_SIZE = 300
	dev (v1_basic)	1.14	
Stacked RNN with Dropout (GRU)	training (v2_adv)	51.59	NUM_STEPS = 35 LEARNING_RATE = 0.0001 HIDDEN_SIZE = 750 VOCAB = 20000 EPOCH = 10 BATCH_SIZE = 20 EMBEDDING_SIZE = 300 NUM_LAYER = 2 DROPOUT = 0.5
Stacked RNN with Dropout (GRU)	dev (v2_adv)	39.75	

Tabelle 2: RNN Modelle mit Hyperparameter

reduziert. Hingegen wurde die ursprüngliche Vokabulargrösse von 10'000 auf 20'000 verdoppelt, um der Korpusgrösse besser Rechnung zu tragen. Dies erhöht natürlich den Schwierigkeitsgrad für das Modell, da nicht mehr so viele UNK-Elemente vorhergesagt werden können und sich auch in der Perplexity niederschlägt.

Aufgrund der unterschiedlichen Datenaufbereitung und des vergrösserten Vokabulars sind die Perplexity-Werte der verschiedenen Modelle nicht direkt vergleichbar. Die Perplexity von 39.75 des Stacked RNN scheint mir aber für die gewählte Vokabulargrösse kein schlechter Wert zu sein, der als neue Baseline für das weitere Tunen verwendet werden könnte. Dank dem Dropout könnte die Anzahl Layer und Zellen stark vergrössert werden, ohne dass man in Gefahr eines Overfitting läuft.

Anderes

Neben einem falschen Verständnis bei der Implementation von einem bidirektionalen RNN (über OLAT geklärt), bekundete ich noch mit dem folgenden Code Probleme. Offenbar sind diese Teile nicht äquivalent und führen zu Tensoren mit falschen Shapes.

```
def cell():             # Example 1
return tf.contrib.rnn.GRUCell(HIDDEN_SIZE)
tf.contrib.rnn.MultiRNNCell( [cell() for _ in range(NUM_LAYERS)])

stacked_gru = tf.contrib.rnn.MultiRNNCell(             [cell] * NUM_LAYERS) # Example 2, not worki
```

Und hier noch der Satz, über den ich am meisten lachen musste: «The Sudan conferred on terrorism as common good .»

Literatur

Baturo, Alexander, Niheer Dasandi und Slava J. Mikhaylov (1. Apr. 2017). «Understanding State Preferences with Text as Data: Introducing the UN General Debate Corpus». In: *Research & Politics* 4.2: S. 1–9.