

Laufzeituntersuchungen in OpenFST

Lena Schiffer

4. Juni 2018

1 Einleitung

Es wurden insgesamt 3600 Tests durchgeführt, bei denen alle Kombinationen der folgenden Parameter ausprobiert wurden. Das Zeitlimit für jeden Test betrug 10s.

```
vector<string> input_words = {"bleibt", "zuständigen", "miüssen",  
    "radioaktiver", "schießen", "niedersBchsischen"};  
vector<bool> lazy_options = {false, true};  
vector<StdVectorFst*> lexicons = {lexicon_small, lexicon_big};  
vector<StdVectorFst*> morphologies = {NULL, rules};  
vector<int> error_model_options = {123, 1, 2, 3, 23};  
vector<int> error_numbers = {1, 2, 3, 4, 5};  
vector<int> nbests = {1, 10, 50};
```

Hier fehlen Tests für ein erweitertes Lexikon (Lexikon + Morphologie) und für die Vorbereitung von Fehlermodell und Lexikon. Da diese sowieso nur für das kleine Lexikon möglich sind, sind diese Tests eher uninteressant.

Das kleinere Lexikon (`lexicon=0`) umfasst 50k Wörter (von Maciej zusammengestellt), das größere Lexikon (`lexicon=1`) 1m Wörter (aus den Asse-Daten).

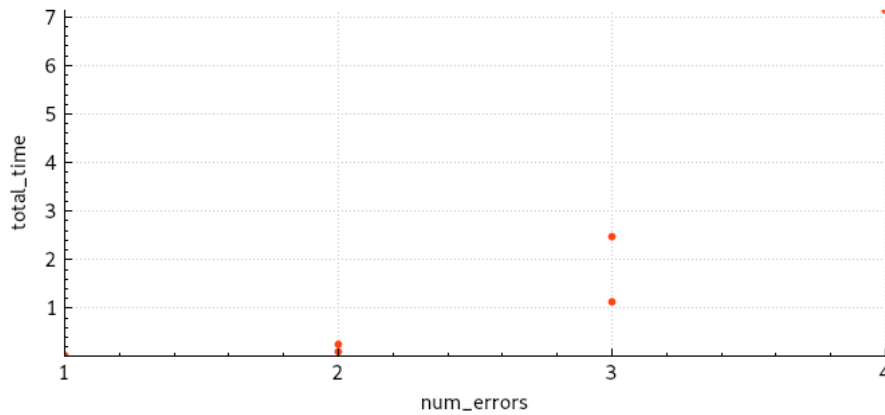
Die `error_model_options` sagen aus, wie viel Zeichen Kontext berücksichtigt werden (123 bedeutet, dass Uni-, Bi- und Trigramme berücksichtigt werden).

`error_numbers` drückt aus, wie viele Edits maximal erlaubt sind.

Die `total_time` setzt sich aus `composition_time` und `search_time` zusammen. Gemessen wird die reine Rechenzeit und sie wird in Sekunden angegeben.

`ArcSort` und Umcodieren der `SymbolTables` fließt nicht mit ein, da das Teil der Vorbereitung ist.

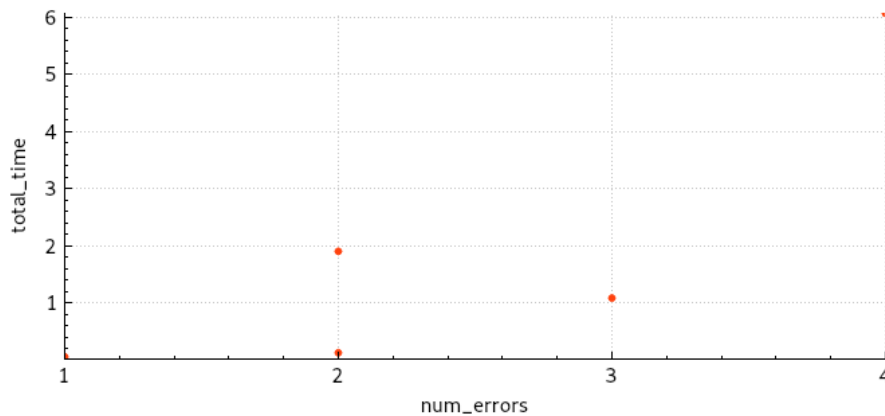
2 Morphologie



lexicon=1, word=bleibt, nbest=1, lazy=0, context=123

Jeweils beim Wert mit höherer `total_time` wurde Morphologie verwendet. Bei bis zu 2 Edits sind die Werte insgesamt nur gering. Bei 3 Edits gibt es größere Unterschiede (1s vs 2,5s). Ab 4 Edits ist das Ergebnis mit Morphologie nicht mehr berechenbar, ab 5 Edits auch ohne Morphologie nicht.

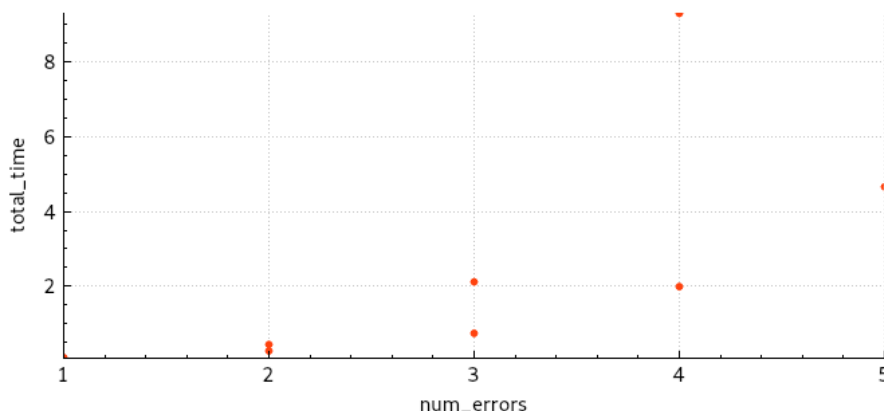
3 Kompositionsmethode



lexicon=1, word=zuständigen, nbest=1, morphology=0

Lazy Composition hat insgesamt eine deutlich höhere Laufzeit als Eager Composition. Obwohl keine Morphologie verwendet wurde, können nur bis zu 2 Fehler berücksichtigt werden (mit Eager Composition bis zu 4 Fehler).

4 Lexikongröße

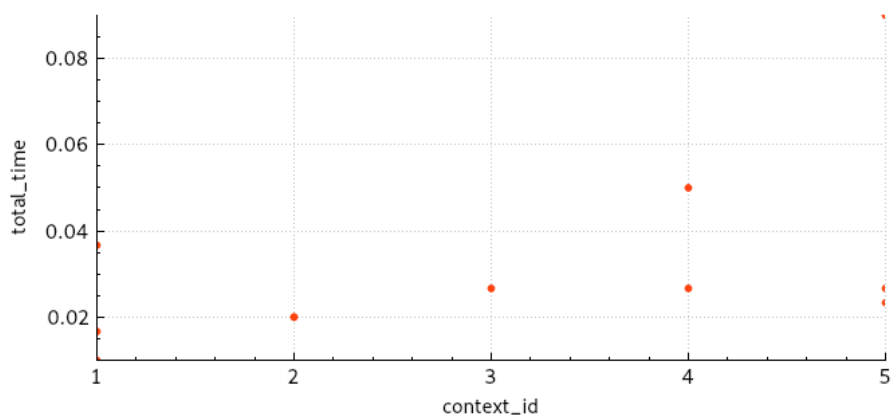


morphology=1, nbest=1, lazy=0, context=123, word=zuständigen
Akzeptable Werte nur mit maximal 2 Edits. Zwar sind die Laufzeiten beim kleineren Lexikon auch mit 3 Edits akzeptabel, doch es sollte das große Lexikon verwendet werden für bessere Korrekturergebnisse.

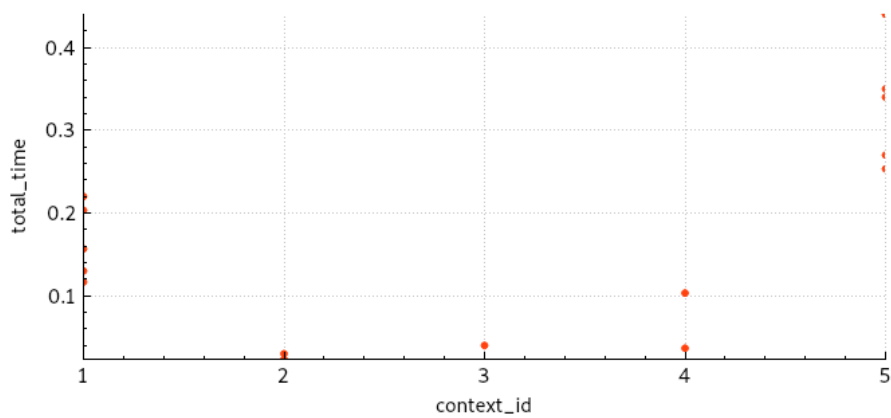
5 Kontextlänge

Fehlermodelle mit verschiedenen Kontexten wurden zusammen mit einer Anzahl Edits von 1 bis 3 getestet. Es wurden alle Wörter berücksichtigt, für die es Ergebnisse gab (`num_states>0`). `context_ids` 1 bis 3 entsprechen Kontext von 1 bis 3 Zeichen. 4 entspricht Kontext 23 und 5 entspricht Kontext 123.

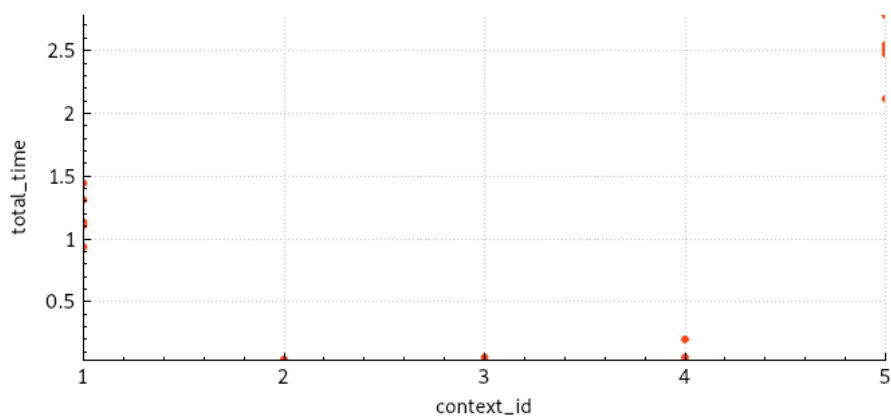
Wie erwartet erhält man die höchste Laufzeit bei Kontext 123 (Unigramme, Bigramme und Trigramme bei der Korrektur berücksichtigt). Unigramme haben wie erwartet den größten Anteil an der Laufzeit. Kontext 2 oder 3 fallen kaum ins Gewicht und haben niedrige Laufzeiten. Bei bis zu 2 Edits sind alle Fehlermodelle akzeptabel, ab 3 Edits nur solche, die keine Unigramm-Substitutionen berücksichtigen.



morphology=1, lexicon=1, num_states>0, lazy=0, num_errors=1

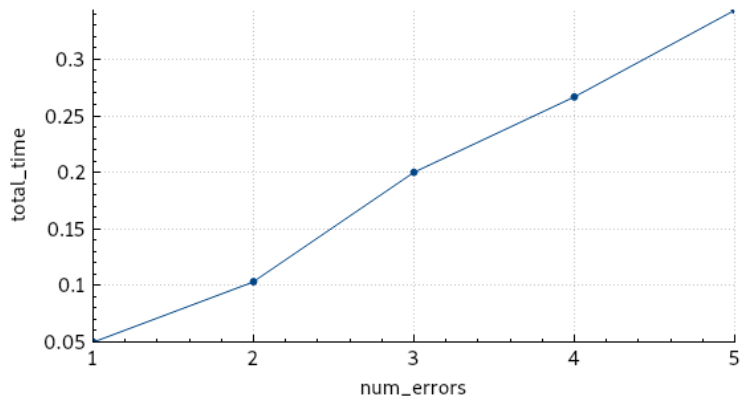


num_errors=2

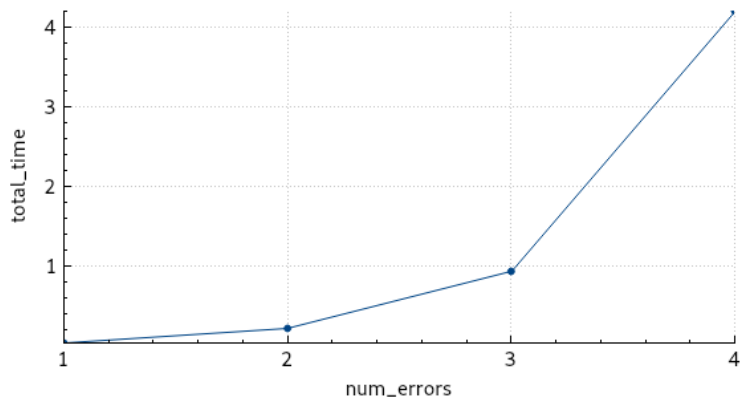


num_errors=3

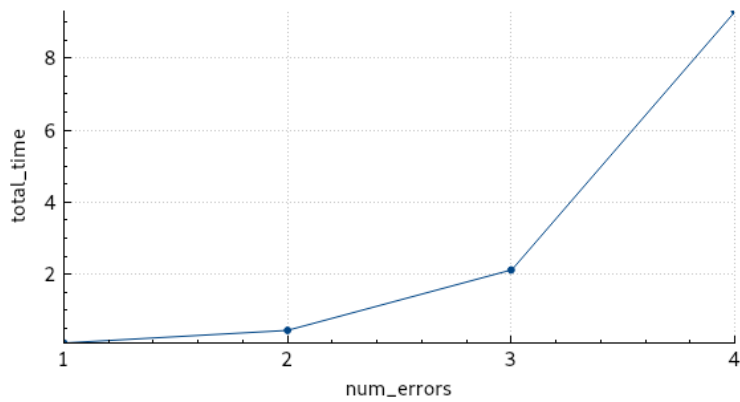
6 Anzahl erlaubter Edits



word=zuständigen, nbest=1, lazy=0, lexicon=1, morphology=1, context=23



context=1



context=123

Die Laufzeit rein kontextbasierter Fehlermodelle steigt etwa linear mit Anzahl der Edits. Durch ein Unigramm-Fehlermodell kommt ein exponentieller Anstieg der Laufzeit hinzu. Bei bis zu 2 Edits sind alle Fehlermodelle akzeptabel. Will man mehr Edits machen, sollte man einen Kontext von 2 bis 3 Zeichen beim Fehlermodell betrachten und die Unigramm-Substitutionen weglassen.

7 Fazit

Die Anzahl der besten Pfade bei der Suche (`nbest`) fällt kaum ins Gewicht (nicht dargestellt).

Lazy Composition hat immer eine schlechtere Laufzeit als Eager Composition. Es ist unklar, woran das liegt, vermutlich aber gibt es ein Problem mit der Implementation der Lazy Composition von OpenFST.

Die Verwendung der Morphologie hat zwar einen nicht unerheblichen Einfluss auf die Laufzeit, doch bei den Tests war eine inakzeptable Laufzeit mit Morphologie auch ohne Morphologie schon inakzeptabel (ab 3 Edits) und umgekehrt.

Die Lexikongröße macht ab 3 Edits einen großen Unterschied aus (von 0,5s auf 2,5s). Da wir für unser Programm ein großes Lexikon nutzen möchten, um die Korrekturqualität zu verbessern, sollte man aber davon ausgehen, dass wir ein großes Lexikon nutzen, und die anderen Parameter entsprechend anpassen.

Den größten Einfluss auf die Laufzeit hat das Fehlermodell, nämlich die Anzahl der Edits und der betrachtete Kontext. Unigramm-Fehlermodelle lassen die Laufzeit mit der Anzahl Edits exponentiell steigen, während kontextbasierte Fehlermodelle sie nur linear ansteigen lassen. Unigramme berücksichtigende Fehlermodelle haben dadurch mit nur maximal 2 Edits eine akzeptable Laufzeit. Kontextbasierte Fehlermodelle sind auch bei 5 Edits noch akzeptabel.

Eine Verbesserung der Laufzeit der Lazy Composition könnte man laut Kay-Michael Würzner mit einer anderen Implementation erreichen, die Tiefensuche verwendet (Bubenzer und Würzner: Joining Composition and Trimming of Finite-State Transducers, 2010).

Um eine Aussage über die Korrekturqualität bei unterschiedlichen Modellen machen zu können, müsste die Anwendung auf längere Abschnitte erweitert werden.