Neural Encoding with Structured Decoding

Pushpendre Rastogi
3^{rd} year CS Phd. Student
pushpendre@jhu.edu

Johns Hopkins University

CLSP Student Seminar, Spring 2016

Outline

1. Improving Neural Network Architectures.

Outline

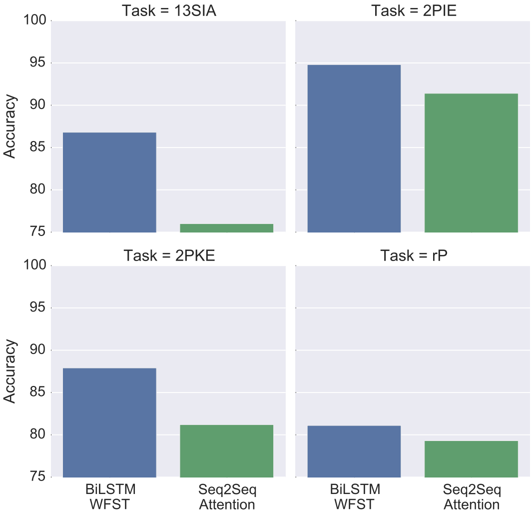
Background: What is the task?

String transduction: Convert an input string to an output string.

Example

- Morphological Transduction:
  - Convert an imperative word in german to its past participle form. `a b r e i b t` ↦ `a b g e r i e b e n`
- Lemmatization:
  - Lemmatize a word in tagalog. `b i n a w a l a n` ↦ `b a w a l`
- Annotate a string:
  - `Bob is a builder` ↦ `Noun Verb Det Noun`

What do we offer?

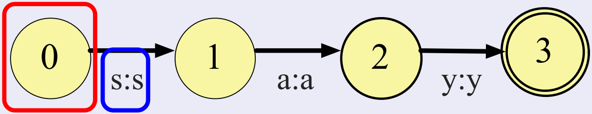


The Idea

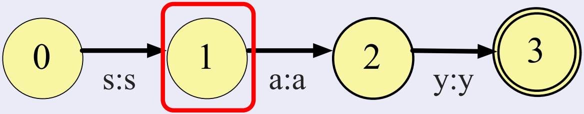
Use a Neural Sequence Encoder to weight the arcs of a Weighted FST.

Background

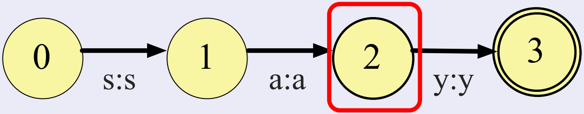
Weighted Finite State Transducers: Deterministic



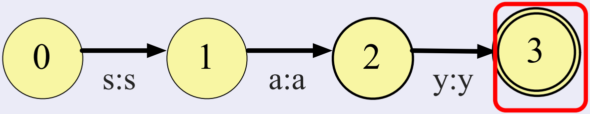
Weighted Finite State Transducers: Deterministic



Weighted Finite State Transducers: Deterministic



Weighted Finite State Transducers: Deterministic

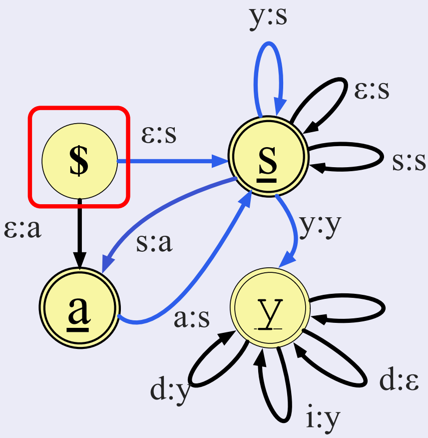


What is a State?

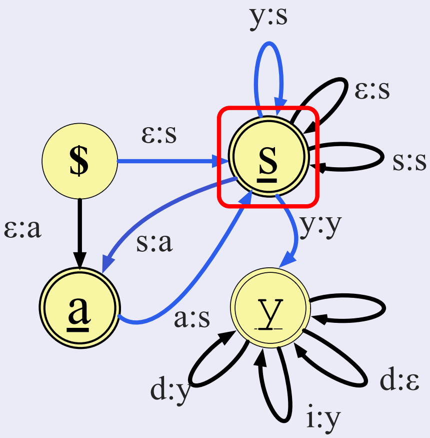
The States of an FST/WFST are its Memory.

Previous Work weights this transducer.

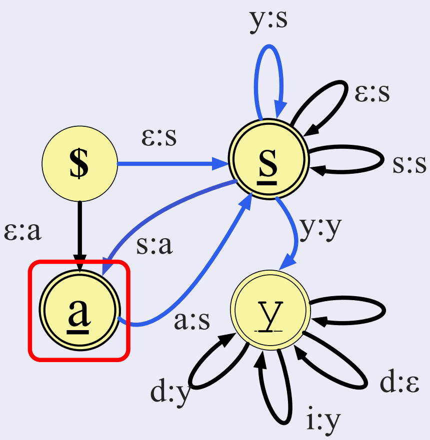
Weighted Finite State Transducers: Non-Deterministic



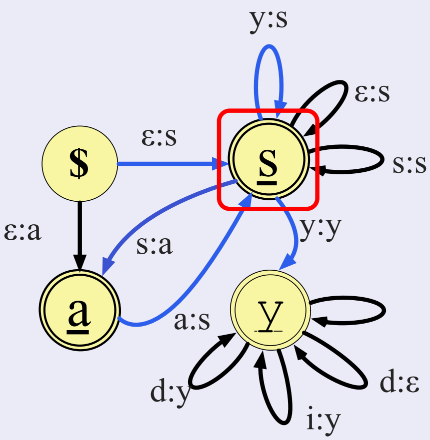
Weighted Finite State Transducers: Non-Deterministic



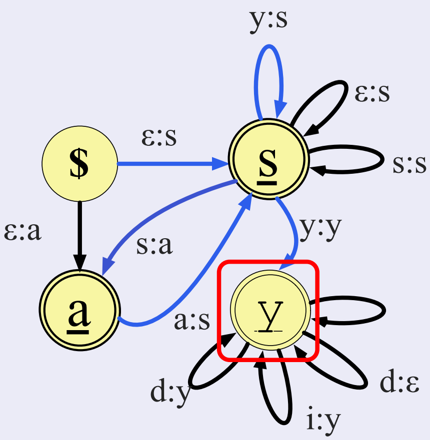
Weighted Finite State Transducers: Non-Deterministic



Weighted Finite State Transducers: Non-Deterministic



Weighted Finite State Transducers: Non-Deterministic



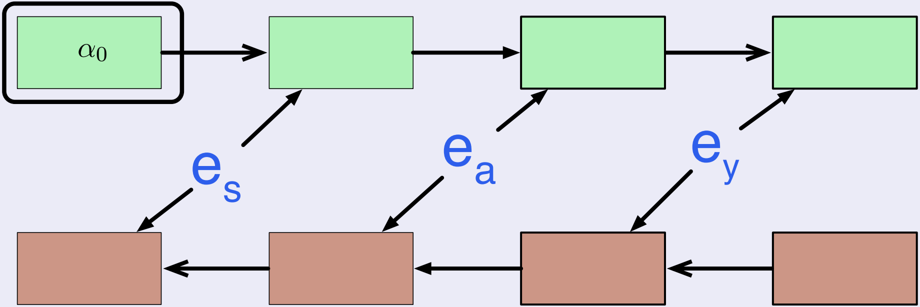
What's in a Path?

A Path is an alignment.

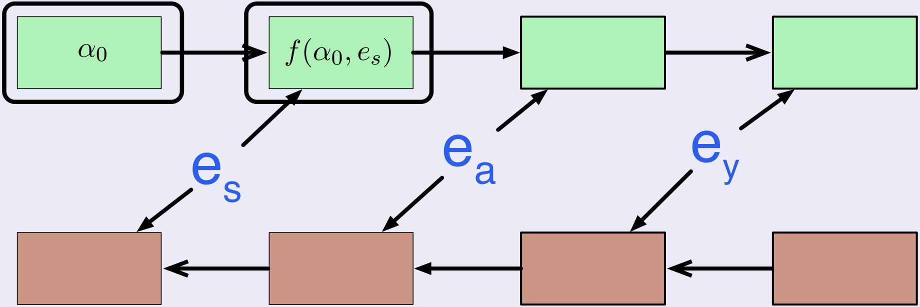
(ϵ:s s:a a:s y:s)	$\mapsto$ say:sass
(ϵ:s s:a a:ϵ y:y)	$\mapsto$ say:say
(ϵ:ϵ s:s a:a y:y)	$\mapsto$ say:say
(ϵ:s s:a a:s y:y)	$\mapsto$ say:sasy

Previous Work weights this transducer.

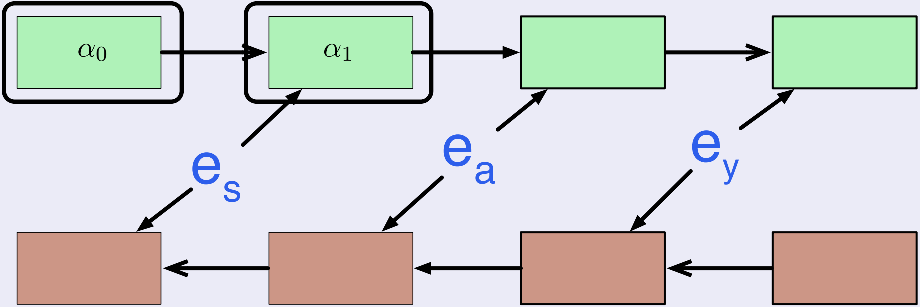
Neural Bi-Directional Sequence Encoder



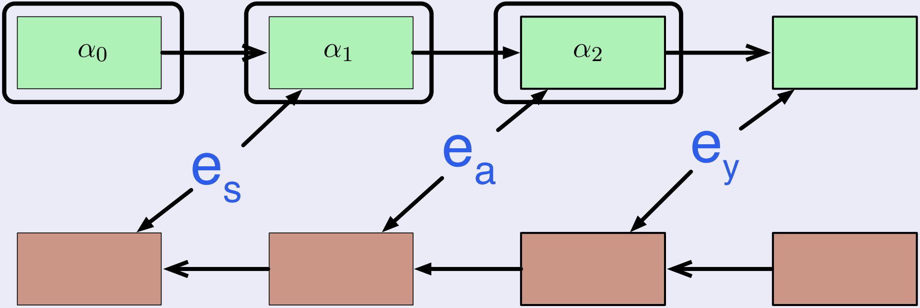
Neural Bi-Directional Sequence Encoder



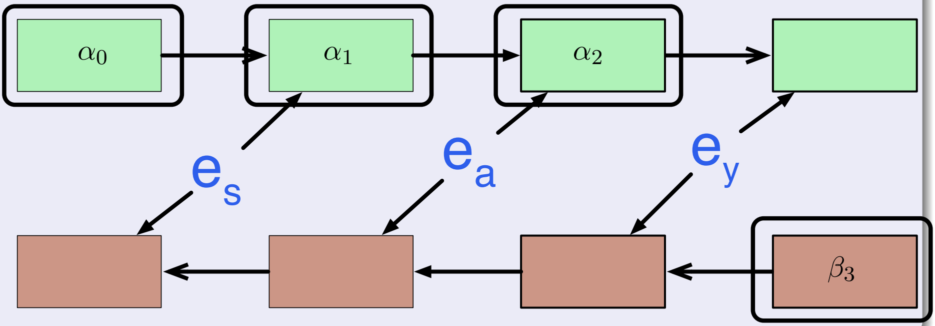
Neural Bi-Directional Sequence Encoder



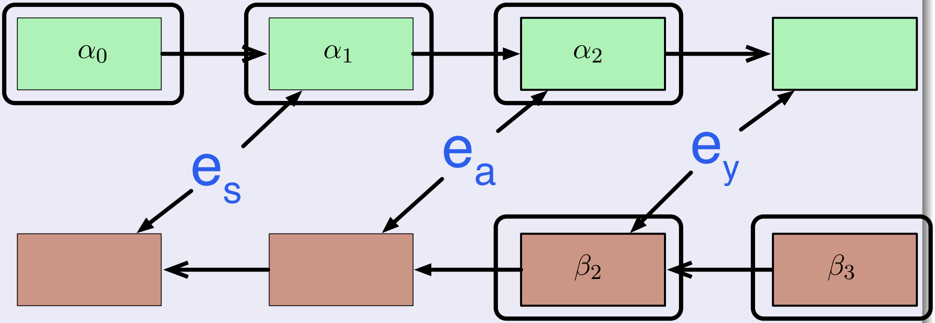
Neural Bi-Directional Sequence Encoder



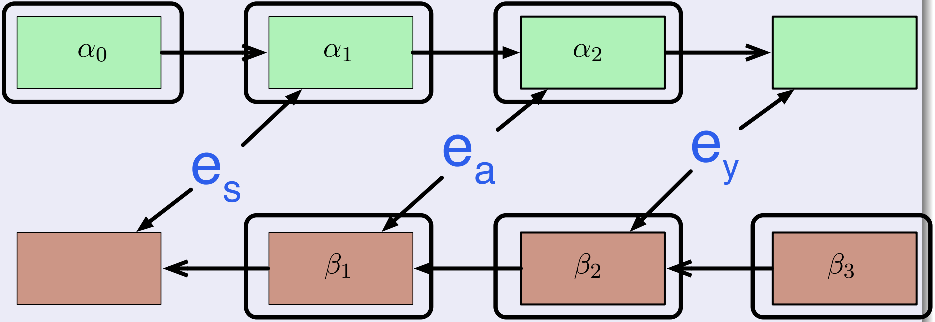
Neural Bi-Directional Sequence Encoder



Neural Bi-Directional Sequence Encoder



Neural Bi-Directional Sequence Encoder



Weighted Finite State Transducers [Moh97, Eis02]

Pros

Cons

Neural Encoders and Decoders [SVL14]

Pros

Cons

JOHNS HOPKINS UNIVERSITY

Weighted Finite State Transducers [Moh97, Eis02]

Pros The states in an FST can be tailored for the task.
Can compute the probability of a string.

Cons

Neural Encoders and Decoders [SVL14]

Pros

Cons

Background: Existing models.

JOHNS HOPKINS UNIVERSITY

Weighted Finite State Transducers [Moh97, Eis02]

Pros The states in an FST can be tailored for the task.
Can compute the probability of a string.

Cons Traditionally arcs weights are linear functionals of arc features.

- ROI on feature engineering may be low.
- The model may become slow if there are too many features.
- The local features may not be expressive enough.

Neural Encoders and Decoders [SVL14]

Pros

Cons

JOHNS HOPKINS UNIVERSITY

Weighted Finite State Transducers [Moh97, Eis02]

Pros The states in an FST can be tailored for the task.
Can compute the probability of a string.

Cons Traditionally arcs weights are linear functionals of arc features.

- ROI on feature engineering may be low.
- The model may become slow if there are too many features.
- The local features may not be expressive enough.

Neural Encoders and Decoders [SVL14]

Pros Produce reasonable results with zero feature engineering.

Cons

Background: Existing models.

Weighted Finite State Transducers [Moh97, Eis02]

Pros The states in an FST can be tailored for the task.
Can compute the probability of a string.

Cons Traditionally arcs weights are linear functionals of arc features.

- ROI on feature engineering may be low.
- The model may become slow if there are too many features.
- The local features may not be expressive enough.

Neural Encoders and Decoders [SVL14]

Pros Produce reasonable results with zero feature engineering.

Cons Require a lot of training data for performance.
Cannot return the probability of a string.

Neural Encoding with Structured Decoding

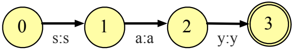


Figure: The automaton I encoding say.

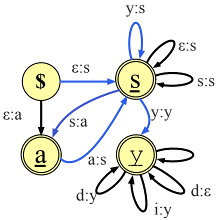


Figure: Transducer F. Only a few of the possible states and edit arcs are shown.

Previous Work weights these transducers

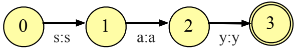


Figure: The automaton I encoding say.

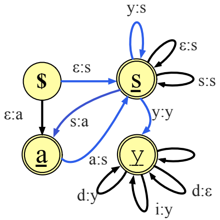


Figure: Transducer F. Only a few of the possible states and edit arcs are shown. Previous Work weights these transducers

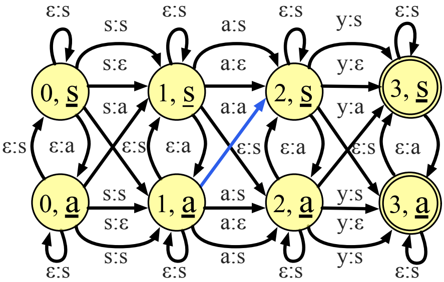


Figure: $G = I \circ F$. Only a few states, but all arcs between them are shown. Our Work weights this transducer.

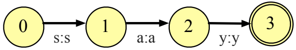


Figure: The automaton I encoding say.

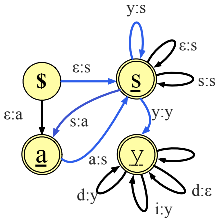


Figure: Transducer F. Only a few of the possible states and edit arcs are shown. Previous Work weights these transducers

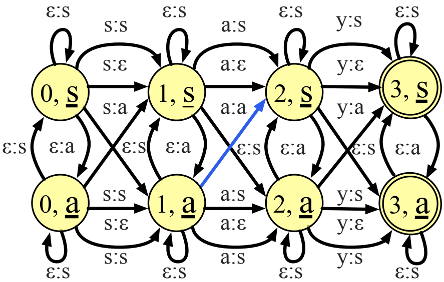
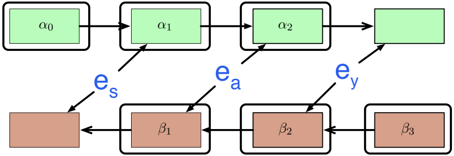


Figure: $G = I \circ F$. Only a few states, but all arcs between them are shown. Our Work weights this transducer.

Why do we do this?

Weighting F ≡	Weighting edits per type.
Weighting G ≡	Weighting edits per token. Neural features encode entire sentence. We get a context dependent output side language model.

Neural Encoding with Structured Decoding



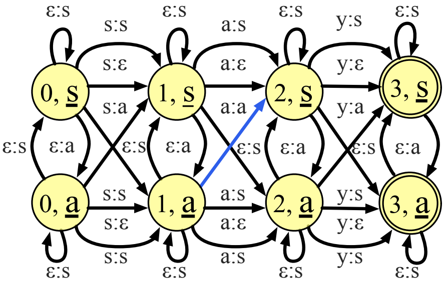
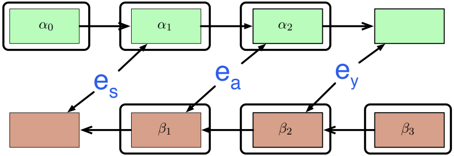


Figure: $G = I \circ F$. Only a few states, but all arcs between them are shown. Our Work weights this transducer.

Neural Encoding with Structured Decoding



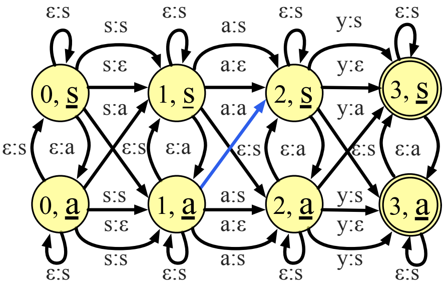
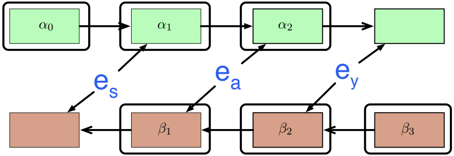
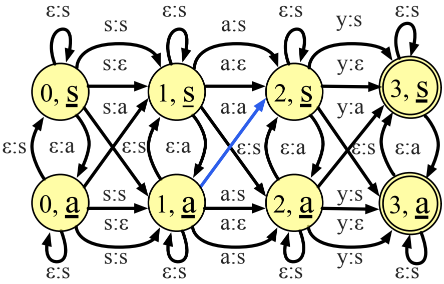


Figure: $G = I \circ F$. Only a few states, but all arcs between them are shown. Our Work weights this transducer.

Idea: Use a BiLSTM to weight the arcs of G.



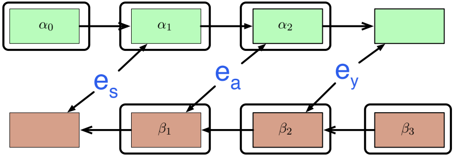


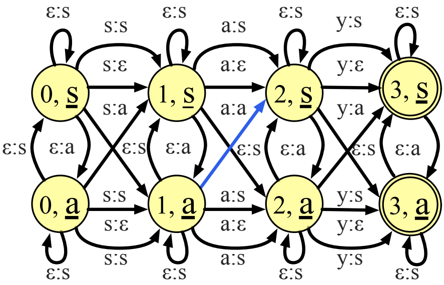
Let $w((1, \underline{a}) \mapsto (2, \underline{s}), a, s) \triangleq \langle v_{\underline{a},s}, (\alpha_2, \beta_1, e_a) \rangle$

$v_{\underline{a},s}$	represents $(\underline{a}, s)$
h may be	the Identity or Relu, . . .

Figure: $G = I \circ F$. Only a few states, but all arcs between them are shown. Our Work weights this transducer.

Idea: Use a BiLSTM to weight the arcs of G.



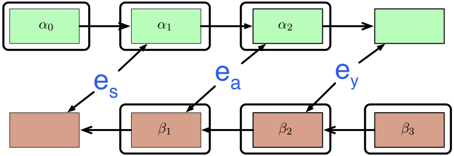


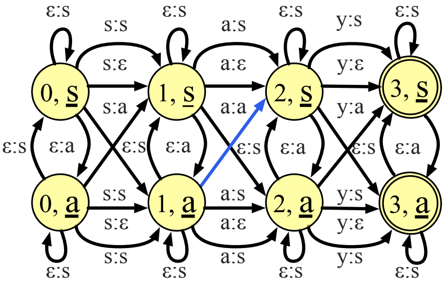
$$\textbf{Let}\quad w((1,\underline{a}) \mapsto (2,\underline{s}), a, s) \triangleq \langle v_{\underline{a},s}, (\alpha_2, \beta_1, e_a)\rangle$$

$v_{\underline{a},s}$	represents $(\underline{a}, s)$
h may be	the Identity or Relu, ...

Figure: $G = I \circ F$. Only a few states, but all arcs between them are shown. Our Work weights this transducer.

Idea: Use a stack of BiLSTM to weight the arcs of G.





Let $w((1, \underline{a}) \mapsto (2, \underline{s}), a, s) \triangleq \langle v_{\underline{a},s}, (\alpha_2, \beta_1, e_a) \rangle$

$v_{\underline{a},s}$	represents $(\underline{a}, s)$
h may be	the Identity or Relu, ...

Figure: $G = I \circ F$. Only a few states, but all arcs between them are shown. Our Work weights this transducer.

Idea: Use a stack of BiLSTM to weight the arcs of G.

Training: SGD of the negative penalized conditional log-likelihood.

Experiments

We conducted experiments on two datasets:

- Morphological Reinflection of German Verbs.
- Lemmatization

Experiments

We conducted experiments on two datasets:

- Morphological Reinflection of German Verbs.

Task	**Input**	**Output**	Training Size	Dev Size	Test Size
13SIA $\mapsto$ 13SKE	abrieb	abreibe	500	1000	1000
2PIE $\mapsto$ 13PKE	abreibt	abreiben	500	1000	1000
2PKE $\mapsto$ z	abreiben	abzurieben	500	1000	1000
rP $\mapsto$ pA	abreibt	abgerieben	500	1000	1000

- Lemmatization

Task	**Input**	**Output**	Training Size	Dev Size	Test Size
Basque	abestean	abestu	4674	584	584
English	activated	activate	3932	492	492
Irish	beathach	beathaigh	1101	138	138
Tagalog	binawalan	bawal	7636	954	954

Experiments

We conducted experiments on two datasets:

- Morphological Reinflection of German Verbs.
- Lemmatization

Model	13SIA	2PIE	2PKE	rP
Moses15	85.3	94.0	82.8	70.8
Dreyer (Backoff)	82.8	88.7	74.7	69.9
Dreyer (Lat-Class)	84.8	93.6	75.7	81.8
Dreyer (Lat-Region)	**87.5**	93.4	**88.0**	**83.7**
BiLSTM-WFST	85.1	**94.4**	85.5	83.0
Model Ensemble	85.8	**94.6**	86.0	**83.8**

Table: Exact match accuracy on Morphological Reinflection.

Model	Basque	English	Irish	Tagalog
Base (W)	85.3	91.0	43.3	0.3
WFAffix (W)	80.1	93.1	70.8	81.7
ngrams (D)	91.0	92.4	96.8	80.5
ngrams + x (D)	91.1	93.4	97.0	83.0
ngrams + x + l (D)	**93.6**	**96.9**	**97.9**	88.6
BiLSTM-WFST	91.5	94.5	**97.9**	**97.4**

Table: Exact match accuracy on Lemmatization.

Experiments: The Learning Curve

Johns Hopkins University

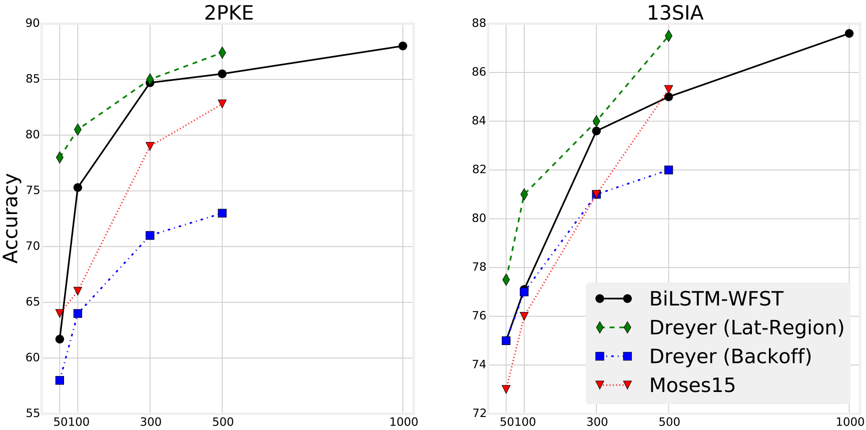
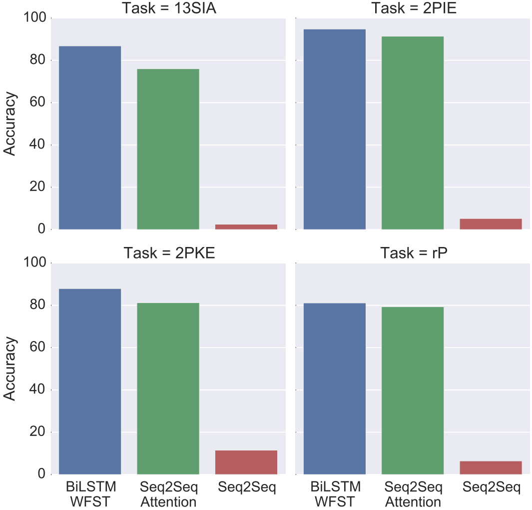
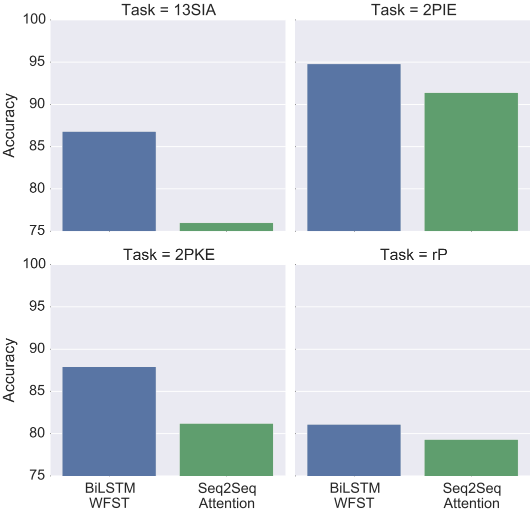


Figure: Best match accuracy on test data Vs. Number of training samples.

Comparison between Sequence-to-sequence based models and the proposed model, on the validation set of morphological re-inflection tasks.



Comparison between Sequence-to-sequence based models and the proposed model, on the validation set of morphological re-inflection tasks.



Outline

Acknowledgements

I collaborated with Ryan Cotterell and Jason Eisner for the work on neural-transducer hybrids. It is the culmination of a lot of earlier unpublished work done with Mo Yu, Dingquan Wang, Nanyun Peng and Elan Hourticolon-Retzler.

During this project I was sponsored by DARPA under the DEFT Program (Agreement FA8750-13-2-0017).

References

Jason Eisner.
Parameter estimation for probabilistic finite-state transducers.
In Proceedings of the ACL, pages 1–8, Philadelphia, July 2002.

Mehryar Mohri.
Finite-state transducers in language and speech processing.
Computational linguistics, 23(2):269–311, 1997.

Ilya Sutskever, Oriol Vinyals, and Quoc Le.
Sequence to sequence learning with neural networks.
In Proceedings of NIPS, 2014.

Extra Slide