Neural Models of Text Normalization

Hao Zhang
Google Research

Overview

- Summary of the problem
- Neural Architectures
- Covering grammars
- Results & Conclusions

The problem

Train a model to verbalize text like the left column as in the right column:

A	<self>
baby	<self>
giraffe	<self>
is	<self>
6ft	**six feet**
tall	<self>
and	<self>
weighs	<self>
150lb	**one hundred and fifty pounds** ← “semiotic class” instance
.	sil

On	<self>
11/11/2016	**november eleventh twenty sixteen**
£1	**one pound**
was	<self>
worth	<self>
$1.26	**one dollar and twenty six cents**
.	sil

Text Normalization as Machine Translation Why Not?

Train a model to translate the sentence in the first row to the one in the second row:

```
A baby giraffe is 6ft tall and weighs 150lb.
A baby giraffe is six feet tall and weighs one hundred and fifty pounds <sil>

On 11/11/2016 £1 was worth $1.26.
On November eleventh twenty sixteen one pound was worth one dollar and twenty six cents <sil>
```

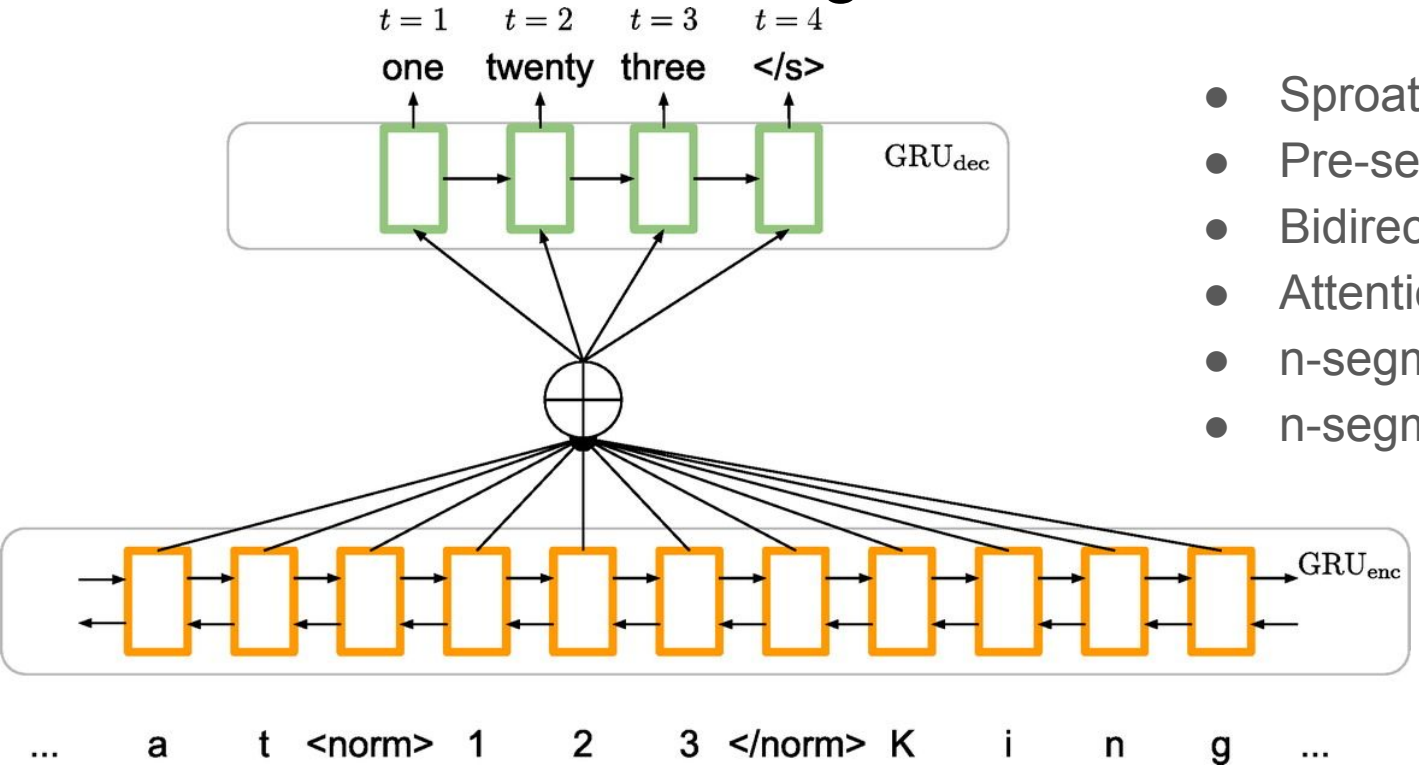
Results of a Transformer Model

Details of the model

We report Sentence Accuracy to in order to compare with our models.

- English:
  - Transformer: 96.53%
  - Our best: 97.75%
- Russian:
  - Transformer: 93.35%
  - Our best: 95.46%
- Main drawback: harder to reconstruct which output token(s) correspond to which input token(s), and thus harder to correct errors.

Our First Model: Sliding Window Context Model



- Sproat and Jaity 2016
- Pre-segmented input
- Bidirectional encoder
- Attention-based decoder
- n-segments to the left
- n-segments to the right

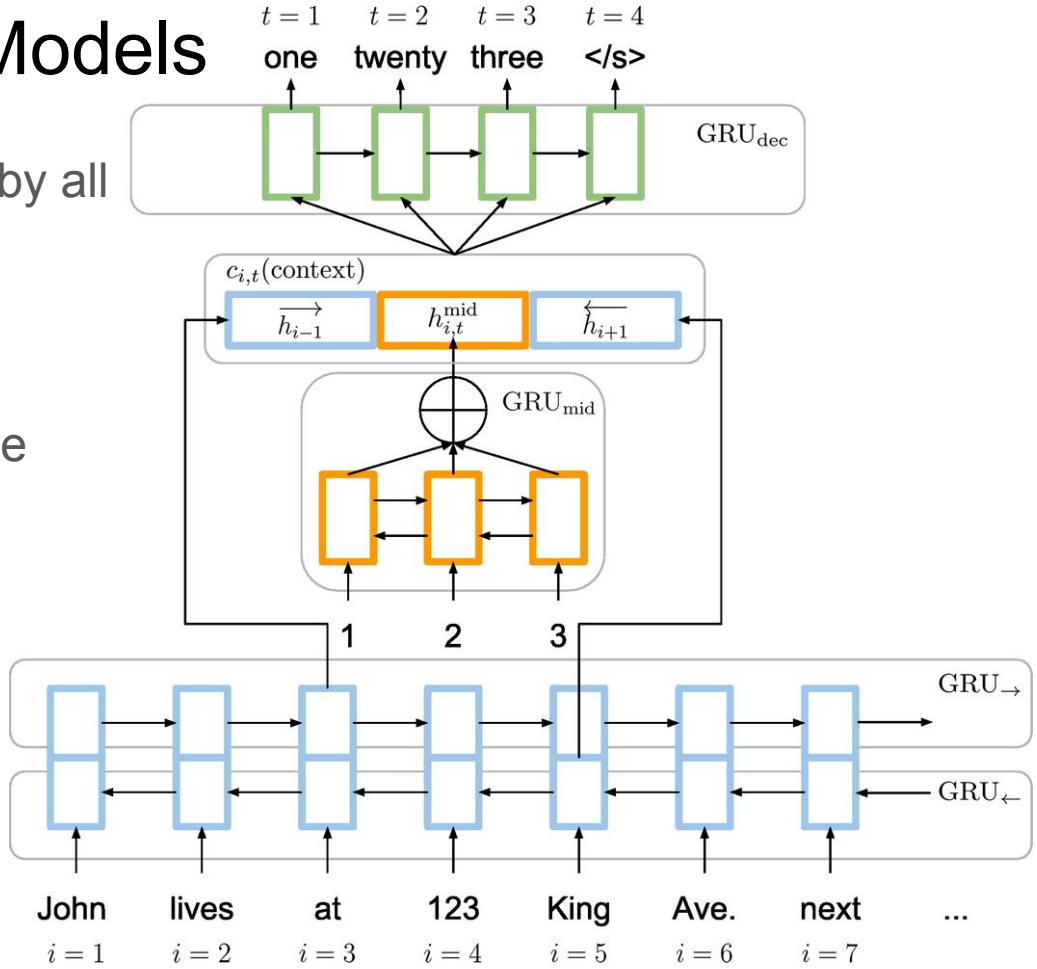
John lives at <norm> 123 </norm> King Ave next to A&P .

Issues of the Sliding Window Model

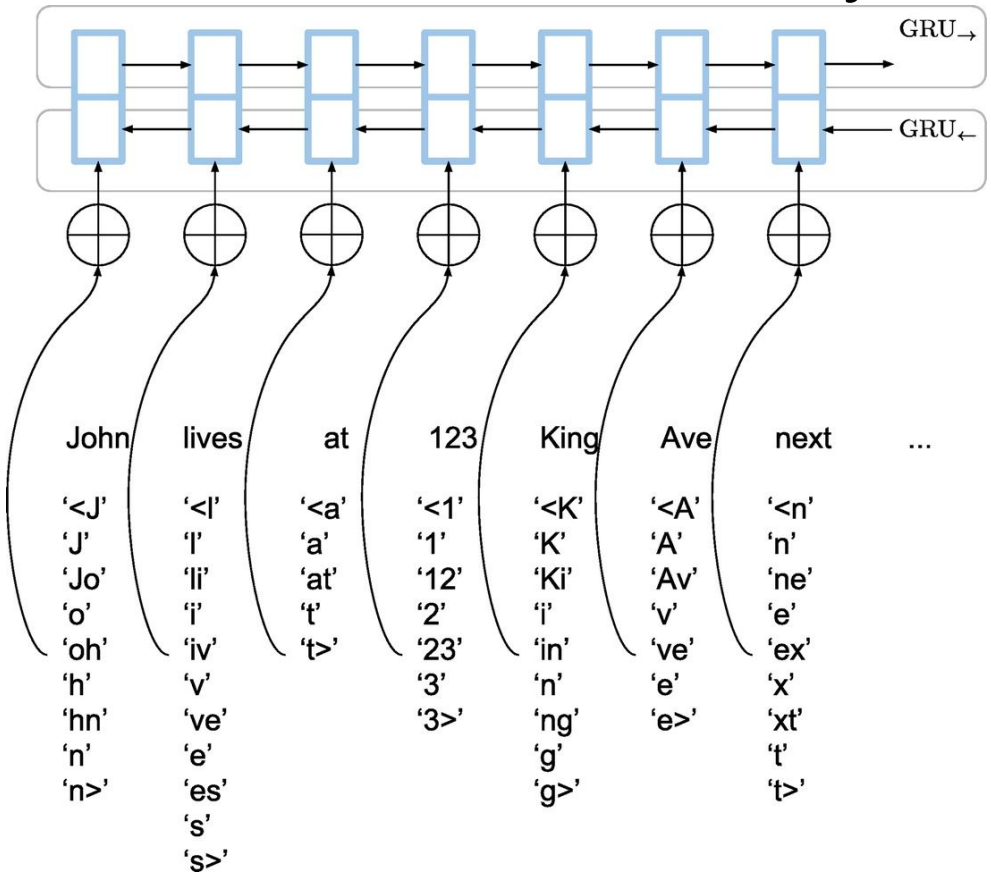
- Limited context
- Repeated encoding computation (constant factor of window size n)
- Assumption of pre-segmented input

(Sentence) Contextual Models

- A sentence context layer shared by all segments.
- Context layer is computed once.
- Context state, instead of context prefix & suffix, is consumed by the decoder.

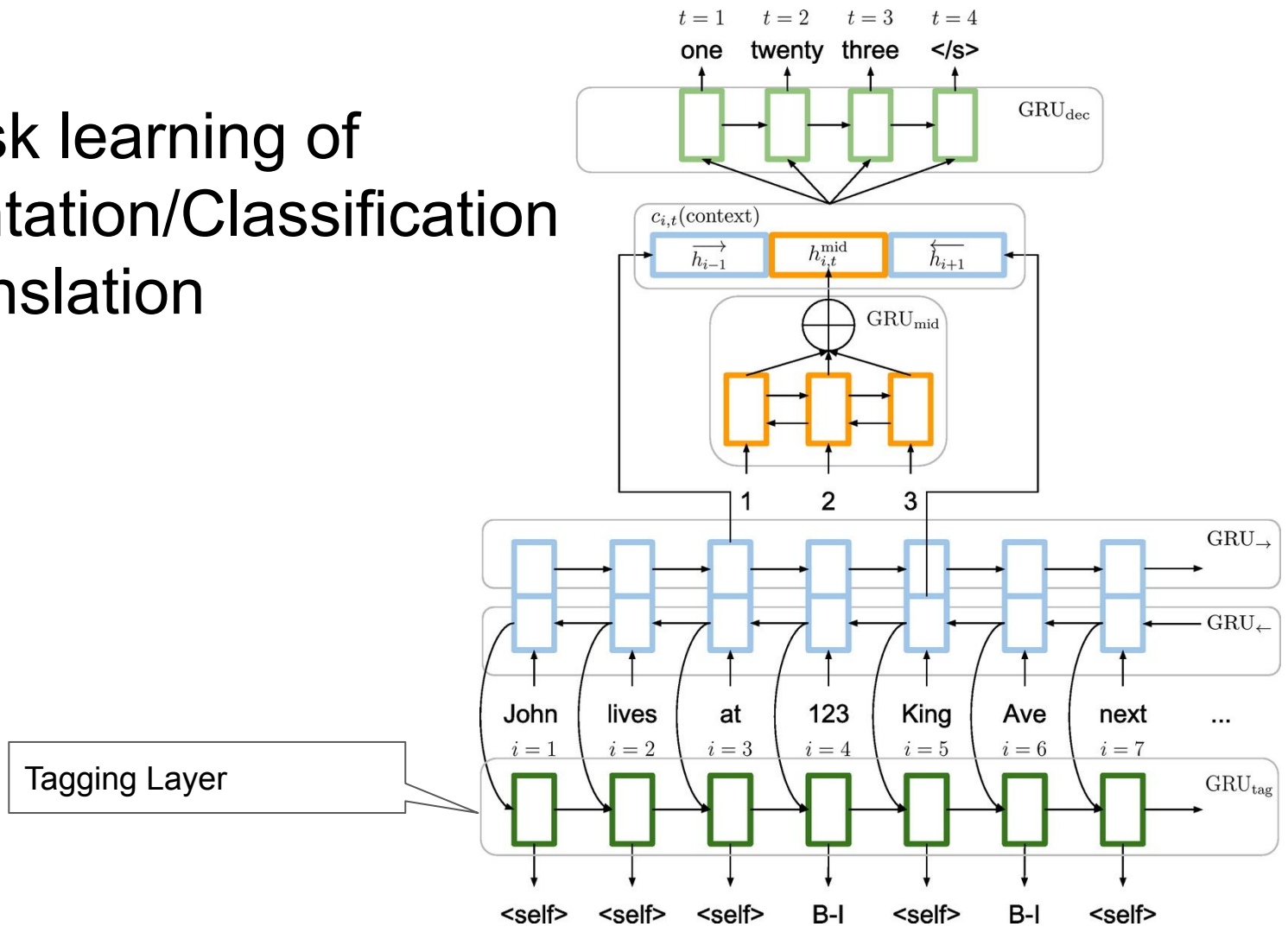


Details of the Context Layer



- Character n-gram embeddings are summed up to represent words
- No OOV
- Sufficient for context representation (disambiguation)
- Efficient

Multi-task learning of Segmentation/Classification and Translation



Results: Comparing Various Contextual Models

Table 3
Results for models of different context representations: Char(acter), W(ord)P(iece), W(ord)F(eature). For the contextual models we specify the tokenization level of the sentential context. *: difference is statistically significant ($p < .05$ on the McNemar test) in comparison with the sliding window baseline. For speed, e.g., "1.4x" means 1.4 times faster.

	Accuracy(Error)			Speed
	All	Semiotic class	Sentence	
English (Standard):				
Sliding window	99.79% (0.21%)	98.20% (1.80%)	97.99% (2.01%)	1.0x
Context (Char)	99.79% (0.21%)	98.20% (1.80%)	97.87% (2.13%)	1.3x
Context (WP)	99.79% (0.21%)	98.35% (1.65%)	97.76% (2.24%)	1.4x
Context (WF)	99.84% (0.16%)*	98.30% (1.70%)	98.20% (1.80%)	1.4x
Russian (Standard):				
Sliding window	99.64% (0.36%)	97.31% (2.69%)	95.61% (4.39%)	1.0x
Context (Char)	99.65% (0.35%)	97.26% (2.74%)	95.70% (4.30%)	1.8x
Context (WP)	99.61% (0.39%)	96.94% (3.06%)*	95.26% (4.74%)	1.8x
Context (WF)	99.62% (0.38%)	97.01% (2.99%)*	95.46% (4.54%)	2.0x

Results: Segmentation-Classification-Translation

Table 5
Results with stacked tagging and normalization models. "+ tag. loss": with multi-task loss of tokenization and semiotic class tagging. "+ tok/tag": with a stacked tokenization and semiotic class tagging model.

	F_1	Acc (Err)		Speed
	All	Segmentation	Sentence	
English (Standard):				
Word-feat.	99.84% (0.16%)	100.00% (0.00%)	98.20% (1.80%)	1.4x
+ tag. loss	99.83% (0.17%)	100.00% (0.00%)	98.12% (1.88%)	1.5x
+ tok./tag.	99.75% (0.25%)	99.26% (0.74%)	97.75% (2.25%)	4.0x
Russian (Standard):				
Word-feat.	99.62% (0.38%)	100.00% (0.00%)	95.46% (4.54%)	2.0x
+ tag. loss	99.65% (0.35%)	100.00% (0.00%)	95.62% (4.38%)	2.2x
+ tok./tag.	99.59% (0.41%)	99.63% (0.37%)	95.46% (4.54%)	5.7x

Unrecoverable Errors and Covering Grammars

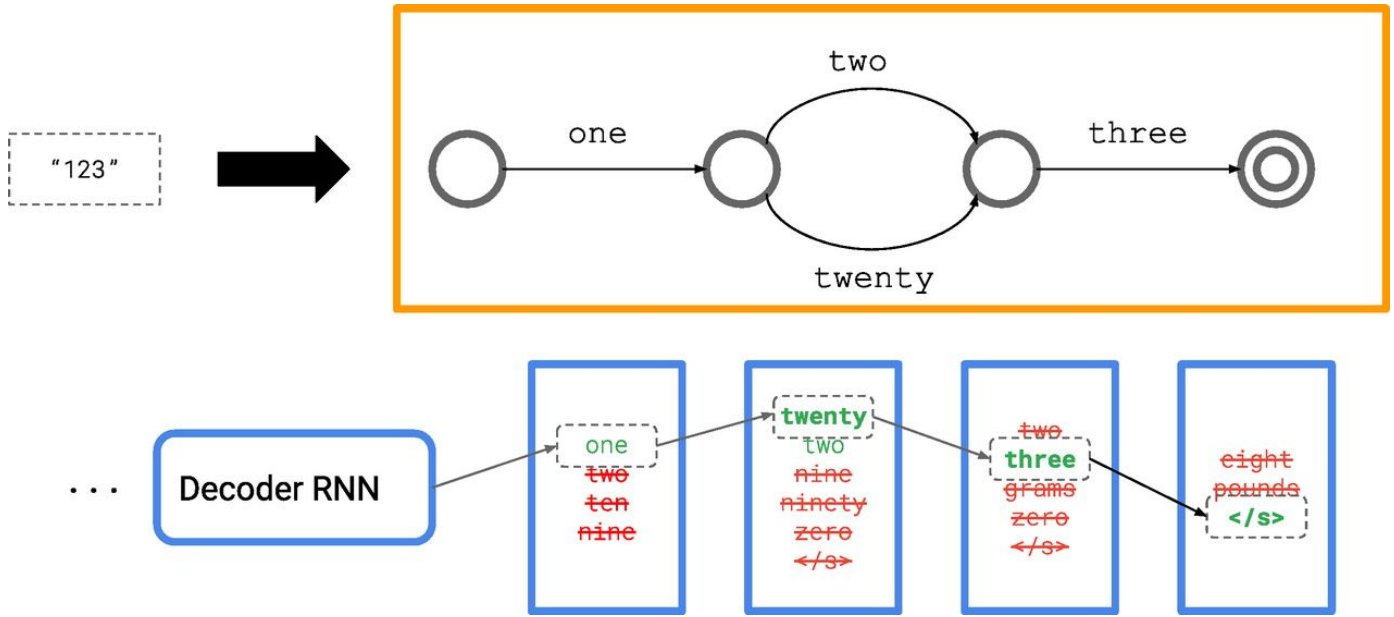
- *€90 million* as *ninety million dollars*
- *3mA as three million liters*
- Covering grammars: lightweight grammars, which enumerate the reasonable possible verbalizations for an input, rather than trying to specify the correct version for a specific instance.

Inducing Covering Grammars from Text

- Gorman and Sproat (2016) reported on a system that can induce a number-name grammar expressed as an FST from a minimal set of training pairs consisting of digit sequences and their verbalizations.
- This approach is extended to more general semiotic classes.

Using Covering Grammars in Seq-to-seq Models

On-the-fly intersection:



Results of Covering Grammars

Table 11
Per-semiotic class error counts with and without covering grammars, covering only classes where there is a difference between using and not using the covering grammar.

	English		Russian	
	Standard	Kaggle	Standard	Kaggle
CARDINAL:				
Best model	1	49	1	73
+ CG decoding	0	15	3	36
DATE:				
Best model	2	24	0	33
+ CG decoding	1	4	0	3
DECIMAL:				
Best model	0	0	0	1
+ CG decoding	0	0	0	7
DIGIT:				
Best model	0	4	0	0
+ CG decoding	0	3	0	0
MEASURE:				
Best model	2	453	2	38
+ CG decoding	1	75	3	29
MONEY:				
Best model	0	18	0	9
+ CG decoding	0	4	1	4
ORDINAL:				
Best model	1	2	0	4
+ CG decoding	0	0	0	2

Conclusions

- We propose an efficient two-stage model for text normalization.
- And its applications in speech: text-to-speech.
- It is suitable in other areas of text transduction, esp., input and output token-level alignment can be induced from data easily.
- It appears to work better than heavy-weight machineries such as Transformer and accommodates point fix mechanisms.
- Covering grammars are effective in reducing unrecoverable errors.
- Open research problem: warranty for elimination of certain type of unrecoverable errors in neural models.
    - Data augmentation, active learning, other ways of constraining the search space