Fast Semantic Extraction Using a Novel Neural Network Architecture

Ronan Collobert & **Jason Weston**

ronan@collobert.com jaseweston@gmail.com

NEC Laboratories America

Summary

I. The Goal: Semantic Role Labeling

II. Previous Work: Parse Trees and SVMs

II. Our Work: End-to-end learning via Neural Networks

Part I

The Goal:

Semantic Role Labeling

Semantic Role Labeling

⋆ Task: Label segments of sentences with semantic roles

[The company]$_{\text{ARG0}}$ [bought]$_{\text{REL}}$ [sugar]$_{\text{ARG1}}$ [on the world market]$_{\text{ARGM-LOC}}$ [to meet export commitments]$_{\text{ARGM-PNC}}$

⋆ Uses: information extraction, call center, search, web crawling. . .

⋆ Need for Speed: lots of data, fast answer for interactive systems

⋆ Previous solutions: use a parse tree, slow

⋆ Our solution: direct mapping

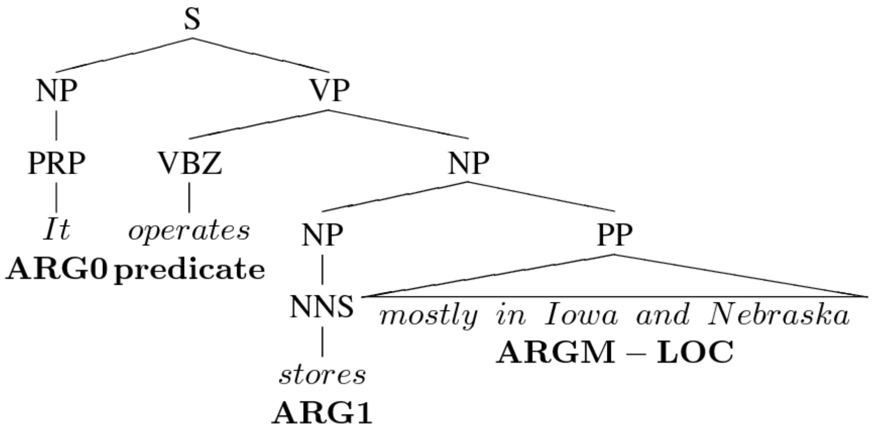
Part II

Previous Work:

Parse Trees and SVMs

ASSERT: State-of-the-Art* (Pradhan et al, 2004)

- ⋆ Run a parser, like Charniak's parser



- ⋆ An SVM predicts for each node of the parse tree whether it has a semantic role or not
- ⋆ If yes, another set of one-vs-rest SVMs classifies the exact role

slow + slow = super slow

*Many other methods exist (see CONLL 2004/2005) – but they have a similar flavor.

ASSERT: Hand Built Features for SVMs

- ⋆ Predicate and POS tag of predicate
- ⋆ Voice: active or passive (hand-built rules)
- ⋆ Phrase type: adverbial phrase, prepositional phrase, . . .
- ⋆ Governing category: Parent node's phrase type(s)
- ⋆ Head word and POS tag of the head word
- ⋆ Position: left or right of verb
- ⋆ Path: traversal from predicate to constituent

More ASSERT features

- ⋆ Predicted named entity class
- ⋆ Word-sense disambiguation of the verb
- ⋆ Verb clustering
- ⋆ Length of the target constituent (number of words)
- ⋆ NEG feature: whether the verb chunk has a ”not” in it
- ⋆ Partial Path: lowest common ancestor in path
- ⋆ Head word replacement in prepopositional phrases (hand-built rules)

Err... Even more ASSERT features...

- ⋆ First and last words and POS in constituents
- ⋆ Ordinal position from predicate + constituent type
- ⋆ Constituent tree distance
- ⋆ Temporal cue words (hand-built rules)
- ⋆ Constituent relative features: 9 features representing phrase type, head word and head word POS for parent and left + right siblings
- ⋆ Dynamic class context: previous node labels
- ⋆ How many pirates exist in the world at the current time

Part III

Our work:

End-to-End Learning with Neural Networks

The Brain Way

We propose a radically different, machine learning, approach:

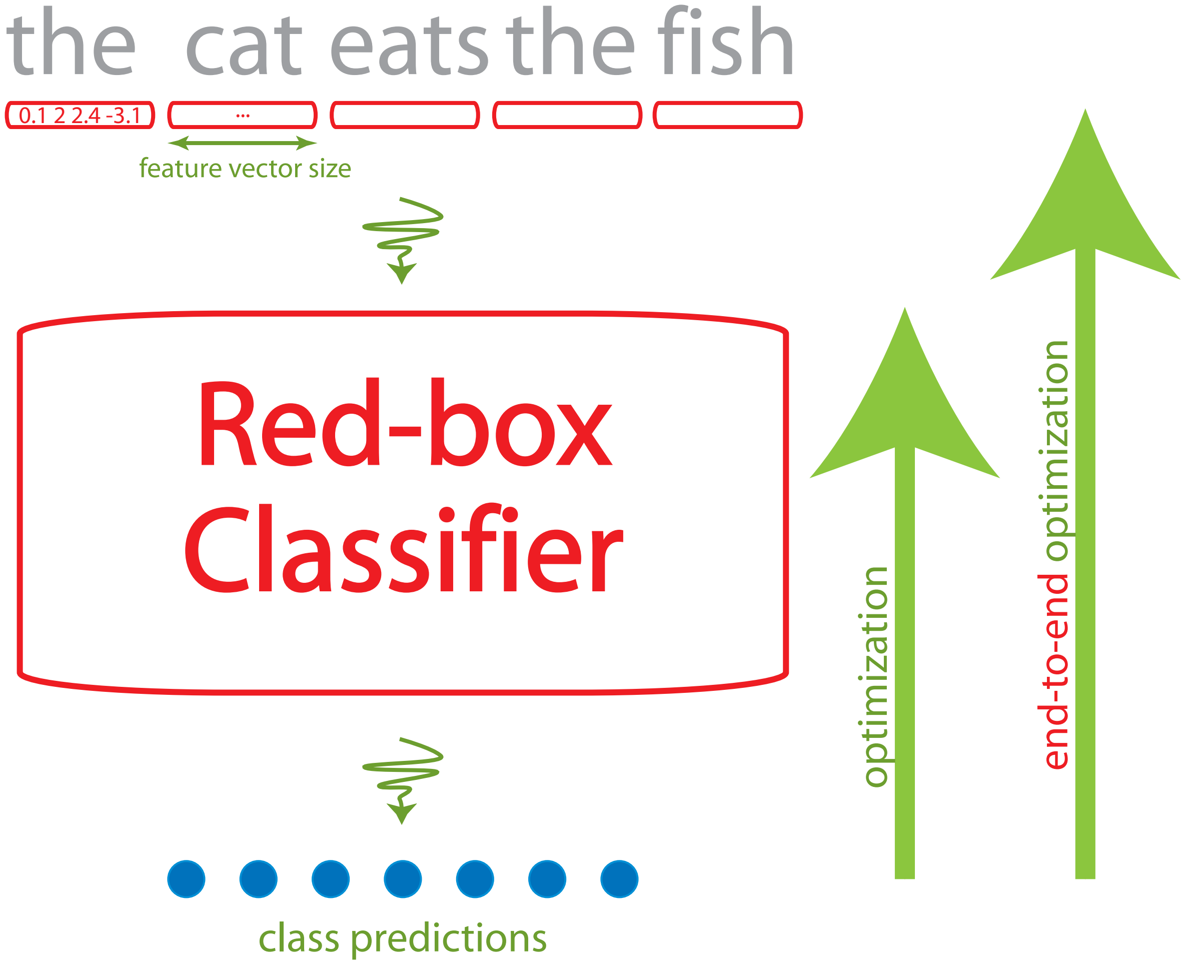
- Avoid building a parse tree. Humans don't need this to talk.
- We try to avoid all hand-built features → monolithic systems.
- Humans implicitly learn these features. Neural networks can too.

End-to-end system
+
Fast predictions (0.02 secs per sentence)

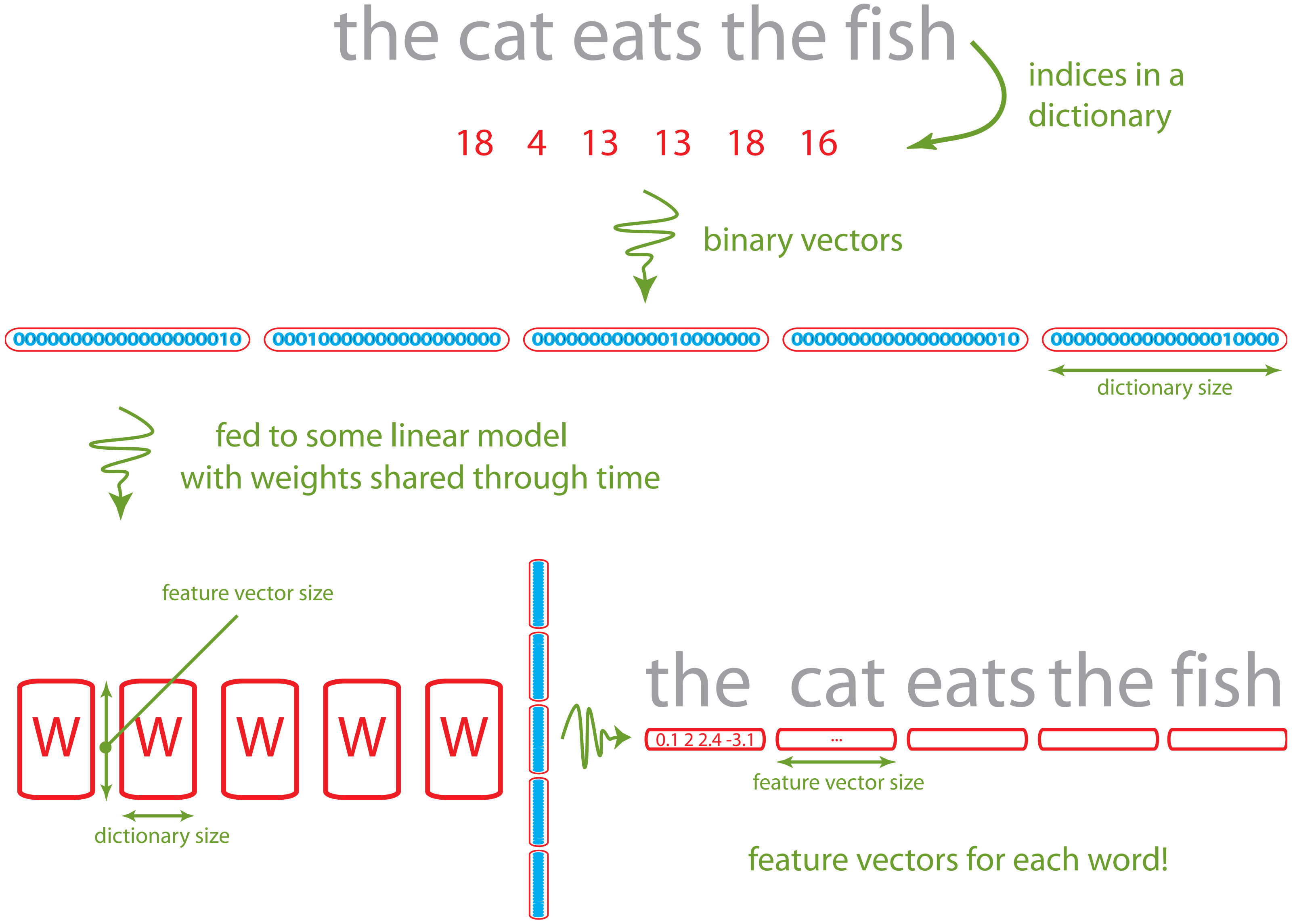
Architecture

- ⋆ Fast: able to handle millions of examples – Neural network
- ⋆ Handle text – 1^{st} layer of network [Bengio et al., 2001]
- ⋆ Tag w.r.t. a predicate – 2^{nd} layer of network [novel contribution]

The Brain Way: End-to-end Learning



1^{st} layer : Words into Vectors: TDNN/CNN



the cat eats the fish

indices in a dictionary

18 4 13 13 18 16

binary vectors

00000000000000000010 00010000000000000000 00000000000010000000 00000000000000000010 00000000000000010000

dictionary size

fed to some linear model
with weights shared through time

feature vector size

W W W W W

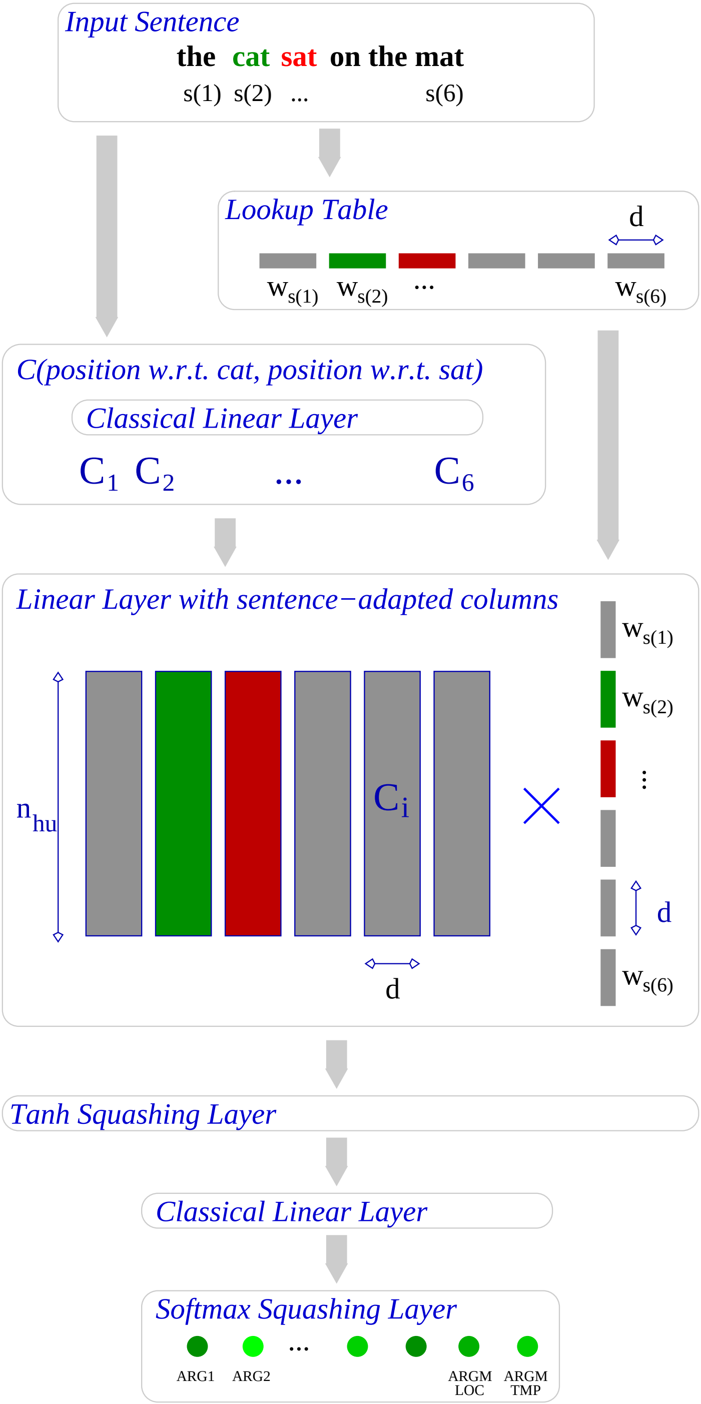
dictionary size

the cat eats the fish

0.1 2 2.4 -3.1 ...

feature vector size

feature vectors for each word!



Input Sentence

the cat sat on the mat

s(1) s(2) ... s(6)

Lookup Table

d

$w_{s(1)}$ $w_{s(2)}$ ... $w_{s(6)}$

C(position w.r.t. cat, position w.r.t. sat)

Classical Linear Layer

C_1 C_2 ... C_6

Linear Layer with sentence−adapted columns

n_{hu}

C_i

d

$w_{s(1)}$

$w_{s(2)}$

⋮

d

$w_{s(6)}$

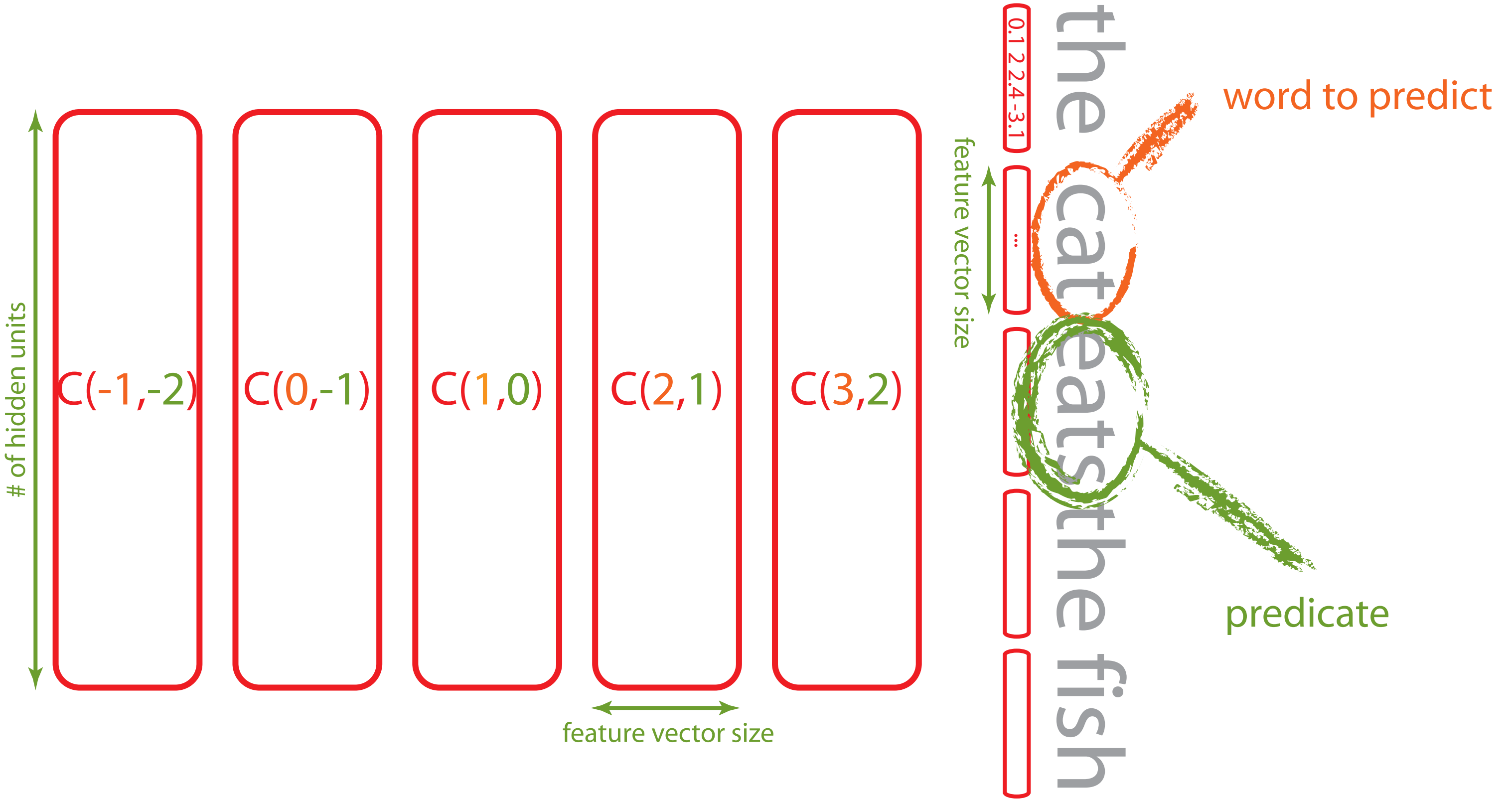
Tanh Squashing Layer

Classical Linear Layer

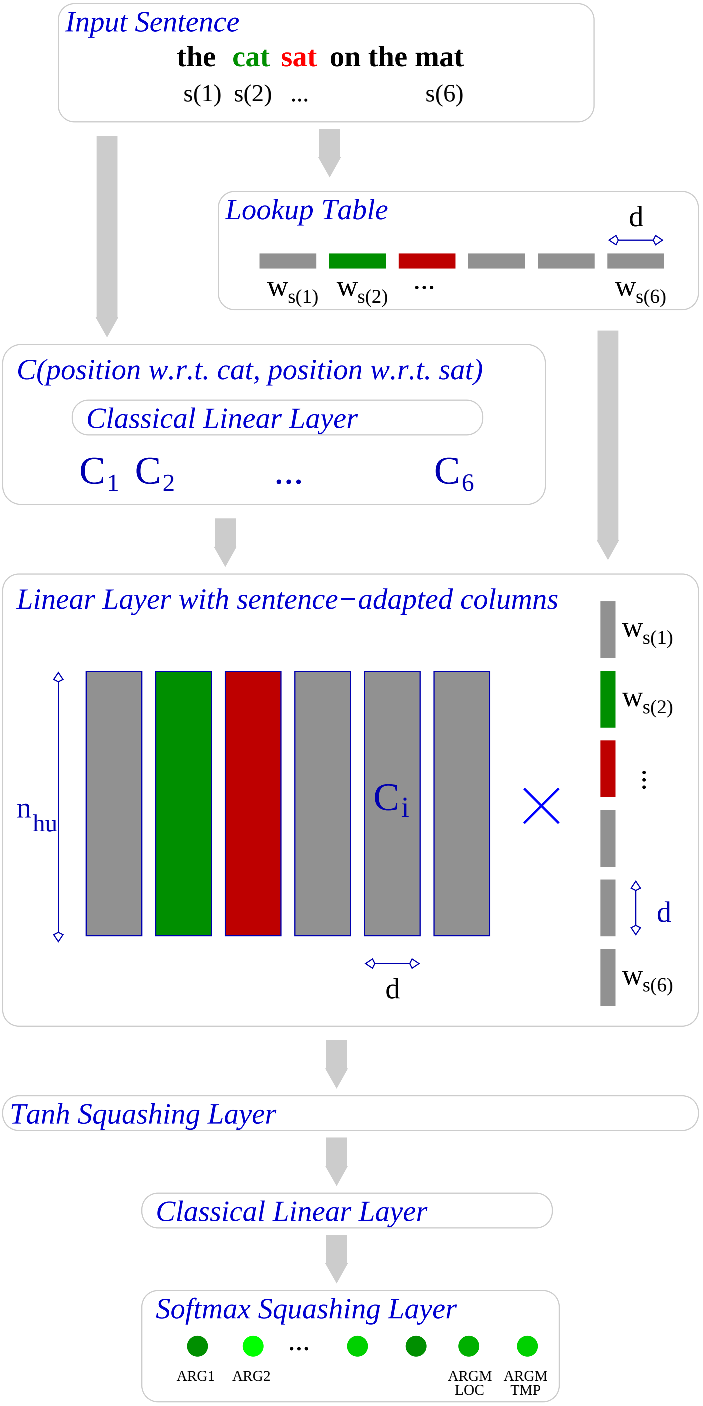
Softmax Squashing Layer

ARG1 ARG2 ... ARGM LOC ARGM TMP

2^{nd} layer : Integrating Word + Verb Positions



C(position w.r.t. word to predict, position w.r.t. predicate) is a function to be chosen


Input Sentence
the cat sat on the mat
s(1) s(2) ... s(6)
Lookup Table
d
$w_{s(1)}$ $w_{s(2)}$ ... $w_{s(6)}$
C(position w.r.t. cat, position w.r.t. sat)
Classical Linear Layer
C_1 C_2 ... C_6
Linear Layer with sentence−adapted columns
n_{hu}
C_i
d
$w_{s(1)}$
$w_{s(2)}$
⋮
d
$w_{s(6)}$
Tanh Squashing Layer
Classical Linear Layer
Softmax Squashing Layer
ARG1 ARG2 ...
ARGM LOC
ARGM TMP

SENNA vs ASSERT

We report experiments on PropBank in the standard train/test split.

ASSERT had no access to a gold standard parse tree.

Measurement	SENNA	ASSERT
Per-word Accuracy	83.64%	83.46%
Per-sentence compute time (secs)	0.02 secs	5.08 secs

Our method is 254x faster than the existing approach.

NOTE: SENNA without 2^{nd} layer trick: 51.3%

Example Output

TRUTH: He camped out at a high-tech nerve center on the floor of [the Big Board, where]$_{\text{ARGM-LOC}}$ [he]$_{\text{ARG0}}$ [could]$_{\text{ARGM-MOD}}$ [watch]$_{\text{REL}}$ [updates on prices and pending stock orders]$_{\text{ARG1}}$.

ASSERT (68.7%): He camped out at a high-tech nerve center on the floor of the Big Board, [where]$_{\text{ARGM-LOC}}$ [he]$_{\text{ARG0}}$ [could]$_{\text{ARGM-MOD}}$ [watch]$_{\text{REL}}$ [updates]$_{\text{ARG1}}$ on prices and pending stock orders.

NN (100%): He camped out at a high-tech nerve center on the floor of [the Big Board, where]$_{\text{ARGM-LOC}}$ [he]$_{\text{ARG0}}$ [could]$_{\text{ARGM-MOD}}$ [watch]$_{\text{REL}}$ [updates on prices and pending stock orders]$_{\text{ARG1}}$.

TRUTH: [United Auto Workers Local 1069, which]$_{\text{ARG0}}$ [represents]$_{\text{REL}}$ [3,000 workers at Boeing's helicopter unit in Delaware County, Pa.]$_{\text{ARG1}}$, said it agreed to extend its contract on a day-by-day basis, with a 10-day notification to cancel, while it continues bargaining.

ASSERT (100%): [United Auto Workers Local 1069, which]$_{\text{ARG0}}$ [represents]$_{\text{REL}}$ [3,000 workers at Boeing's helicopter unit in Delaware County, Pa.]$_{\text{ARG1}}$, said it agreed to extend its contract on a day-by-day basis, with a 10-day notification to cancel, while it continues bargaining.

NN (89.1%): [United Auto Workers Local 1069, which]$_{\text{ARG0}}$ [represents]$_{\text{REL}}$ [3,000 workers at Boeing's helicopter unit]$_{\text{ARG1}}$ [in Delaware County]$_{\text{ARGM-LOC}}$, Pa., said it agreed to extend its contract on a day-by-day basis, with a 10-day notification to cancel, while it continues bargaining.

Final Comments

- Downloadable Software + demo + test results on the WSJ at:

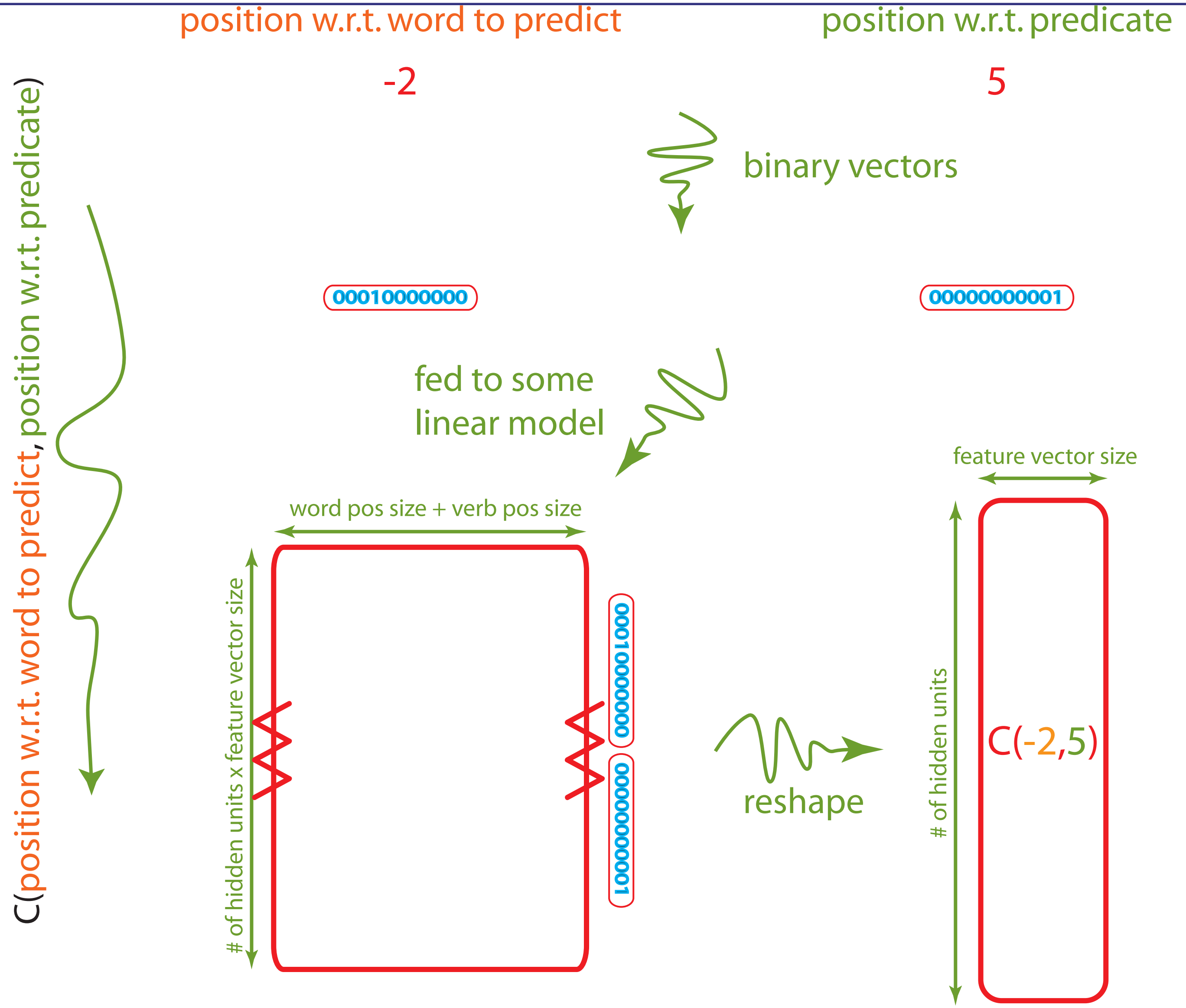
  `http://ml.nec-labs.com/software/senna`

- Open Post-Doc position @ NEC Princeton.

  Speak to me or Ronan Collobert if you're interested.

Thanks!

Extra I: Integrating Word and Verb Positions



position w.r.t. word to predict

position w.r.t. predicate

-2

5

binary vectors

00010000000

00000000001

fed to some linear model

word pos size + verb pos size

# of hidden units x feature vector size

C(position w.r.t. word to predict, position w.r.t. predicate)

reshape

feature vector size

# of hidden units

C(-2,5)

Extra II: Loss Functions

Currently our architecture is designed to label on a per-word basis, while existing systems perform a segmentation process, and then label segments.

We do not optimize our model for the same criteria, but can use the same metrics.

We measured argument classification accuracy, by post-processing our per-word tags to form a majority vote over segments using the parse tree. This gives 83.18% accuracy for our network when we suppose the predicate must also be identified, and 80.53% for ASSERT.

Extra III: Parsing Speed

Even though some parsers effectively exhibit linear behavior in sentence length [Ratnaparkhi et al., 1997], fast statistical parsers such as [Henderson et al., 2004] still take around 1.5 seconds for sentences of length 35 in tests that we made.