

# Natural Language Understanding, Generation, and Machine Translation

## Lecture 29: Unsupervised Parsing

Based on material from Frank Keller

Pasquale Minervini  
[p.minervini@ed.ac.uk](mailto:p.minervini@ed.ac.uk)  
March 27th, 2024

The Story so Far

Unsupervised Parsing

Unsupervised Parsing via Constituency Tests

Grammaticality Model

Results

Reading: Cao et al. (2020)

# The Story so Far

---

- We have seen how **encoder-decoder models** can be used for syntactic parsing: the input is a sequence of words, the output is a sequence of labeled brackets.
- This approach can use LSTMs or transformers, decoding can use beam search or CYK.
- An alternative is to model **parser transitions**, i.e., the actions that are required to build a syntax tree (Kim et al., 2019).  
Not discussed in this course.

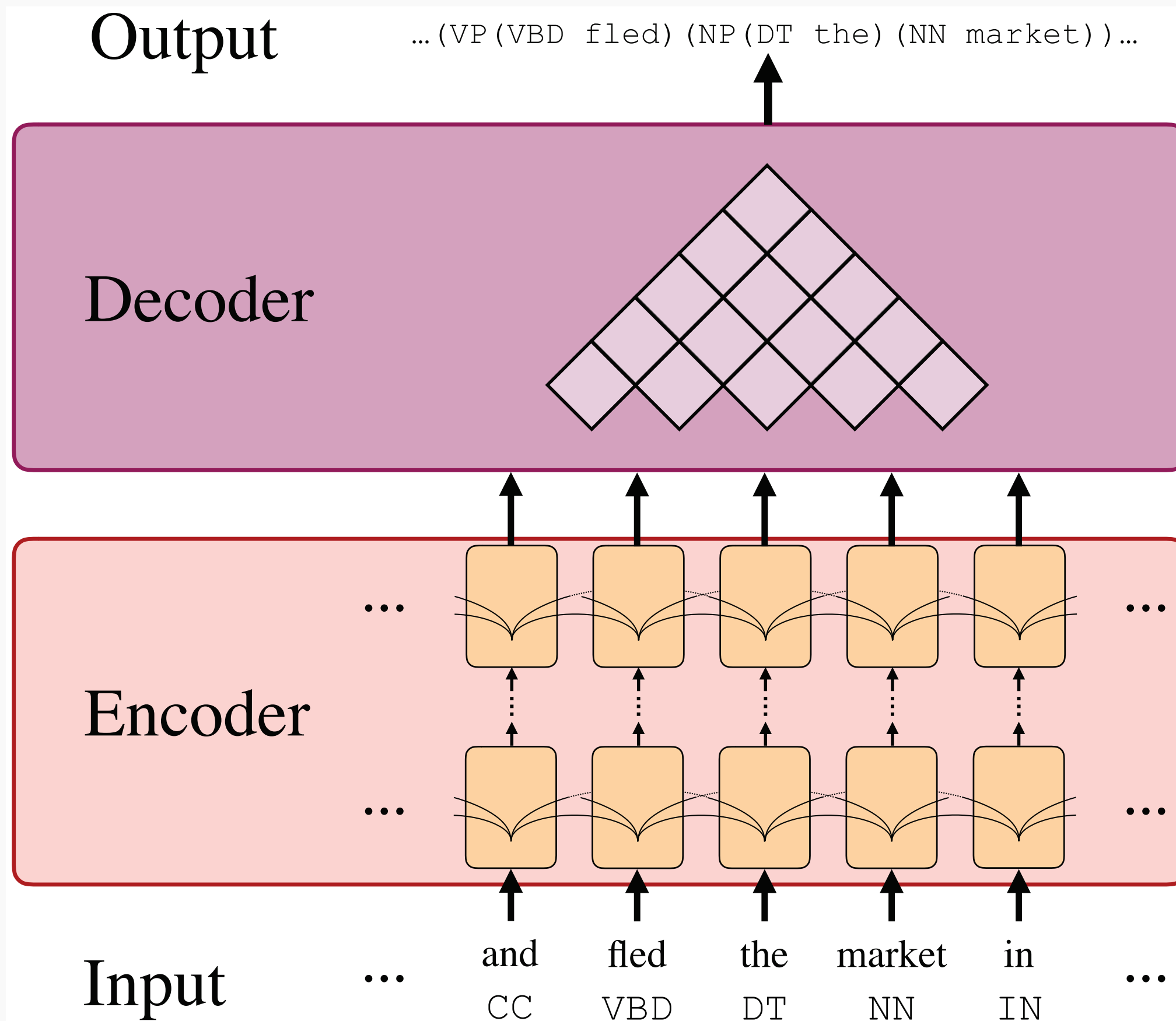
# Neural Parsing

- We have seen how **encoder-decoder models** can be used for syntactic parsing: the input is a sequence of words, the output is a sequence of labeled brackets.
- This approach can use LSTMs or transformers, decoding can use beam search or CYK.
- An alternative is to model **parser transitions**, i.e., the actions that are required to build a syntax tree (Kim et al., 2019).  
Not discussed in this course.

But what if we don't have **syntactically labeled training data**?

Can we still infer syntactic structure if we have lots of text?

# Neural Encoder-Decoder Parsing



# Unsupervised Parsing

---

# Unsupervised Parsing

Goal of unsupervised parsing: induce syntactic structure from **unlabeled text**:

- this is an old problem in the literature on formal grammars (Gold's theorem);
- human children learn the syntax of their native language with little supervision;
- we have treebanks for a few dozen languages (out of the world's around 6,000);
- unsupervised parsing can be used as the first stage in constructing larger treebanks.

But: Unsupervised parsing is hard! SotA F-score is around 60, compared to  $>95$  for supervised parsing.



# Unsupervised Parsing

Key insights for grammar induction:

- exploit the idea behind linguistic constituency tests;
- a sequence of words is a constituent if it appears in the context specific for this constituent;
- long constituents often have short equivalents (pro-forms), e.g., a noun phrase can be replaced by a pronoun;
- the constituency of pro-forms is easy to discover, they are often single words;
- we will use constituency test to discover which **spans** (sequences of words) should be **phrases**.

# Unsupervised Parsing

The insights on the previous page were first used for unsupervised parsing by Klein and Manning (2002).

Their **Constituent Context Model** (CCM) was the first unsupervised parsing model that beat the right-branching baseline for English.

Today, we will look at a neural incarnation of this idea. Cao et al.'s (2020) model combines constituency tests with **pre-trained language models** such as BERT.

# Unsupervised Parsing via Constituency Tests

---

# Constituency Tests

**Example:** *by midday , [ the london market ] was in full retreat*

A: *by midday ,*

B: *the london market*

C: *was in full retreat*

where B is the **span of words being tested for constituency.**

Cao et al. (2020) use the following constituency tests:

Name	Applied to “A [ B ] C”	Example
Clefting	it {is, was} B that A C	<i>it {is, was} the london market that by midday , was in full retreat</i>
Coordination	A B and B C	<i>by midday , the london market and the london market was in full retreat</i>
Substitution	A {it, ones, did so} C	<i>by midday , {it, ones, did so} was in full retreat</i>
Front Movement	B , A C	<i>the london market , by midday , was in full retreat</i>
End Movement	A C B	<i>by midday , was in full retreat the london market</i>

Now try this with a non-constituent (e.g., *market was*)! You will get ungrammatical strings.

# Unsupervised Parsing via Constituency Tests

General approach proposed by Cao et al. (2020):

- **constituency test:** take a sentence, modify it via a transformation (e.g., replace a span with a pronoun);
- check if the result is grammatical using a neural grammaticality model;
- produce a tree for the sentence by aggregating the results of constituency tests over spans;
- select the tree with the highest score;
- add to this **refinement:** alternate between improving the tree and improving the grammaticality model.

# Constituency Tests

**Transformation function:**  $c : (\text{sent}, i, j) \mapsto \text{sent}'$

takes the span  $i$  to  $j$  of a sentence and outputs a new sentence.

**Judgment function:**  $g : \text{sent} \mapsto \{0, 1\}$

judges whether the resulting sentence is grammatical or not.

- Assumption: the span is a constituent if the transformed sentence is grammatical.
- We need a set of transformations (manually specified) and a grammaticality model (learned).
- Constituency tests provide the model with an inductive bias.
- Note that not all constituency tests that linguists use are based on transformations (e.g., semantic preservation).

# Parsing Algorithm

**Key idea:** score each span in the sentence and then choose the tree with the best total score.

Let  $g_\theta : \text{sent} \mapsto [0, 1]$  be the grammaticality model with parameters  $\theta$ .

We score each span by averaging the grammaticality judgments over the constituency tests  $\mathcal{C}$ :

$$s_\theta(\text{sent}, i, j) = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} g_\theta(c(\text{sent}, i, j))$$

Then, we score each tree by summing the scores of its spans and choose the highest scoring binary tree via CYK:

$$t^*(\text{sent}) = \operatorname{argmax}_{t \in T(\text{len}(\text{sent}))} \sum_{(i,j) \in t} s_\theta(\text{sent}, i, j)$$

$T(\text{len}(\text{sent}))$  is the set of binary trees with  $\text{len}(\text{sent})$  leaves.

# Grammaticality Model

---



# Initializing the Grammaticality Model

- To train the grammaticality model, we need a corpus of grammatical and ungrammatical sentences as training data.
- However, this would constitute a form of supervision.
- So instead we train the grammaticality model using a **real/fake task**.
- Given an unlabeled corpus of sentences and a set of **corruptions**, the task is to predict whether a sentence is real or corrupted.
- The grammaticality model is not trained from scratch, but by fine-tuning RoBERTa, a variant of BERT.
- 5 million sentences from English Gigaword serve as the unlabeled training corpus.

# Initializing the Grammaticality Model

Cao et al. use the following set of corruptions:

Name	Description
Shuffle	Choose a random subset of words in the sentence and randomly permute them.
Swap	Choose two words and swap them.
Drop	Choose a random subset of words in the sentence and drop them.
Span Drop	Choose a random contiguous span of words and drop it.
Span Movement	Choose a random contiguous span of words and move it to the front or back.
Bigram	Generate a sentence of the same length using a bigram language model trained

The grammaticality model must determine whether a given sentence is real or corrupted.

# Refining the Grammaticality Model

Refinement procedure:

1. Using the span-based algorithm, parse a batch  $B$  of sentences.
2. Use these trees as pseudo-gold labels to update the span judgments.
3. Repeat for the next batch of sentences.

# Refining the Grammaticality Model

Refinement procedure:

1. Using the span-based algorithm, parse a batch  $B$  of sentences.
2. Use these trees as pseudo-gold labels to update the span judgments.
3. Repeat for the next batch of sentences.

**Updating span judgments:** For each sentence, minimize binary cross-entropy (included in predicted tree or not):

$$\sum_{(i,j) \in t^*(\text{sent})} \log(s_{\theta}(\text{sent}, i, j)) + \sum_{(i,j) \notin t^*(\text{sent})} \log(1 - s_{\theta}(\text{sent}, i, j))$$

$s_{\theta}(\text{sent}, i, j)$  only depends on the grammaticality model, so we are optimizing its parameters: increase the grammaticality judgments of constituent, decrease the judgments of non-constituents.

# Results

---

# Comparison with other Unsupervised Parsers

Model	PTB F1	
	Mean	Max
PRPN <sup>†</sup> (Shen et al., 2018)	37.4	38.1
URNNG (Kim et al., 2019b)	–	45.4
ON <sup>†</sup> (Shen et al., 2019)	47.7	49.4
Neural PCFG <sup>†</sup> (Kim et al., 2019a)	50.8	52.6
DIORA (Drozdov et al., 2019)	–	58.9
Compound PCFG <sup>†</sup> (Kim et al., 2019a)	55.2	60.1
Left Branching	8.7	
Balanced	18.5	
Right Branching	39.5	
Ours (before refinement)	48.2	
Ours (after refinement)	<b>62.8</b>	<b>65.9</b>
Oracle Binary Trees	84.3	

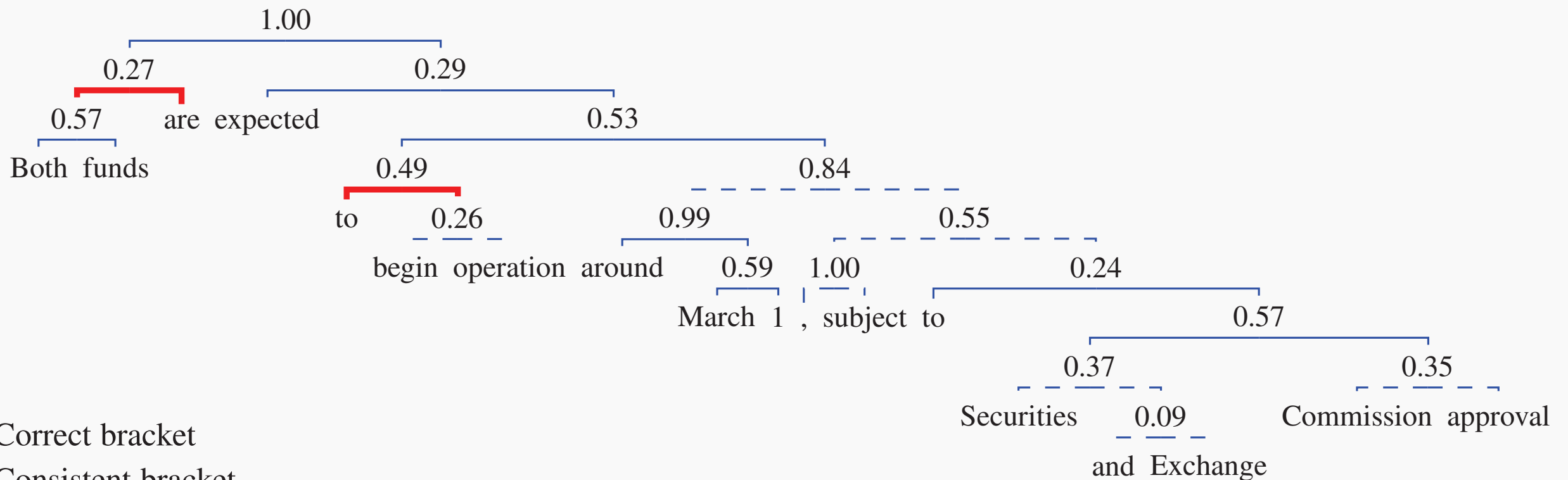
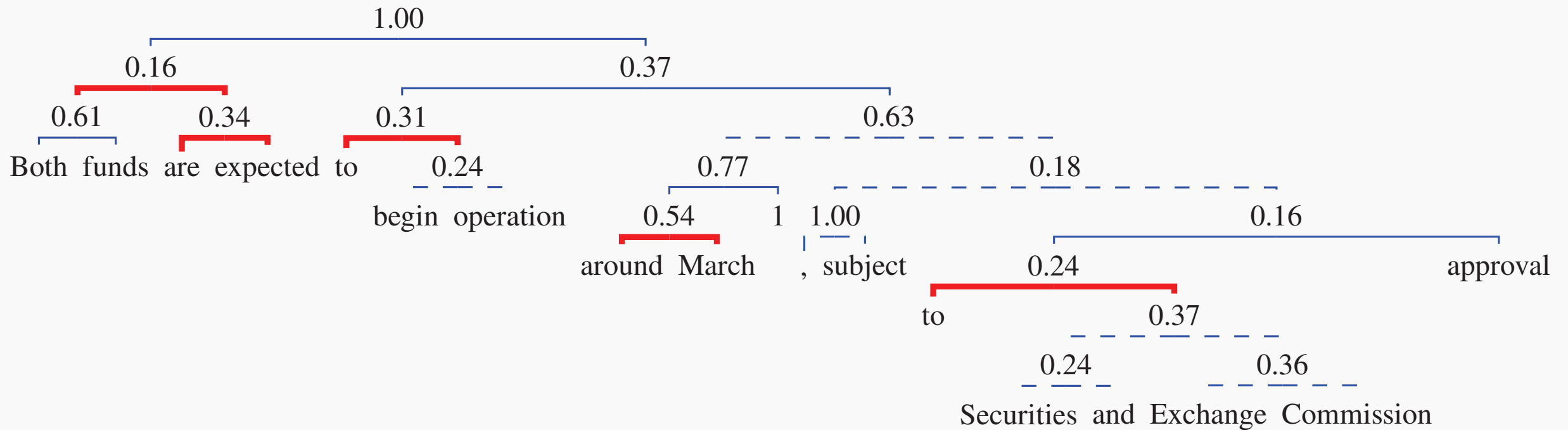
# Combination with Unsupervised RNNG

**Unsupervised RNNG** is a variant of Recurrent Neural Network Grammar proposed by Kim et al. (2019).

URNNG can be trained using the induced trees of another unsupervised parser, followed by fine tuning with an LM objective.

Model	PTB F1	
	Initial (Max)	+URNNG
PRPN	47.9	51.6
ON	50.0	55.1
Neural PCFG	52.6	58.7
Compound PCFG	60.1	66.9
Ours (after refinement)	65.9	<b>71.3</b>
Supervised Binary RNNG	71.9	72.8

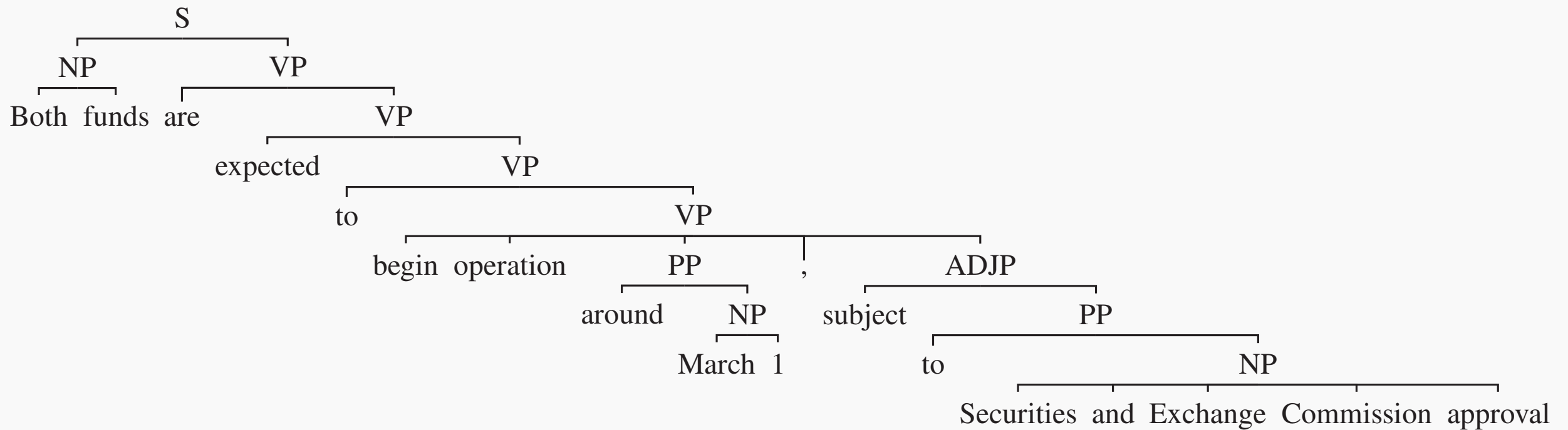
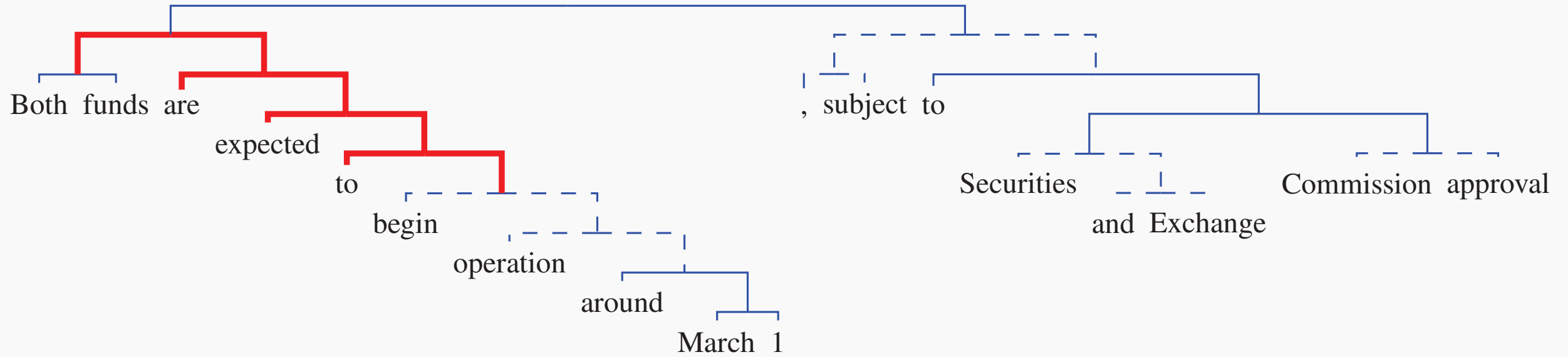
# Analysis of the Output: Before and After Refinement



- Correct bracket
- ⋯ Consistent bracket
- ⊗ Crossing bracket



# Analysis of the Output: After Refinement+URNNG and Gold



- Correct bracket
- ⋯ Consistent bracket
- Crossing bracket

# Analysis of Model Behavior

Cao et al. provide a detailed analysis of the behavior of their model:

- recall by constituency label
- analysis by constituency test
- common mistakes
- example trees

This is something to emulate!

# Summary

- We don't have treebanks for most of the world's language, so we can't train supervised parsers.
- Instead, use unsupervised parsing models, which learn syntax trees from unlabeled text.
- Use constituency tests from linguistic theory: transformation applied to a sentence span that preserves grammaticality.
- Requires a model to judge grammaticality. Obtained by fine-tuning RoBERTa.
- Select the parse tree that maximizes the grammaticality scores of its spans.
- Performs well, but requires with refinement (self-training); can be combined with Unsupervised RNNG.

# References

- Cao, Steven, Nikita Kitaev, and Dan Klein. 2020. Unsupervised parsing via constituency tests. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 4798–4808.
- Kim, Yoon, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. 2019. Unsupervised recurrent neural network grammars. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*. Minneapolis, MN, pages 1105–1117.
- Klein, Dan and Christopher Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, pages 128–135.