Neural architectures for dependency parsing

Marco Kuhlmann

Department of Computer and Information Science

LINKÖPING UNIVERSITY

This work is licensed under a
Creative Commons Attribution 4.0 International License.

Learning problems in dependency parsing

- Learning a greedy transition-based dependency parser amounts to learning the transition classifier.

  Chen and Manning (2014), Kiperwasser and Goldberg (2016)

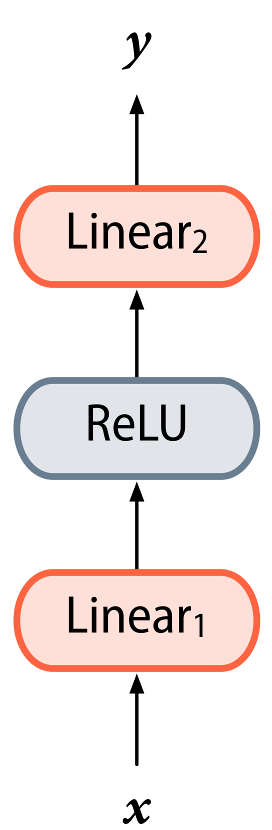
- Learning an arc-factored graph-based dependency parser amounts to learning the arc scores.

  Kiperwasser and Goldberg (2016), Dozat and Manning (2017)

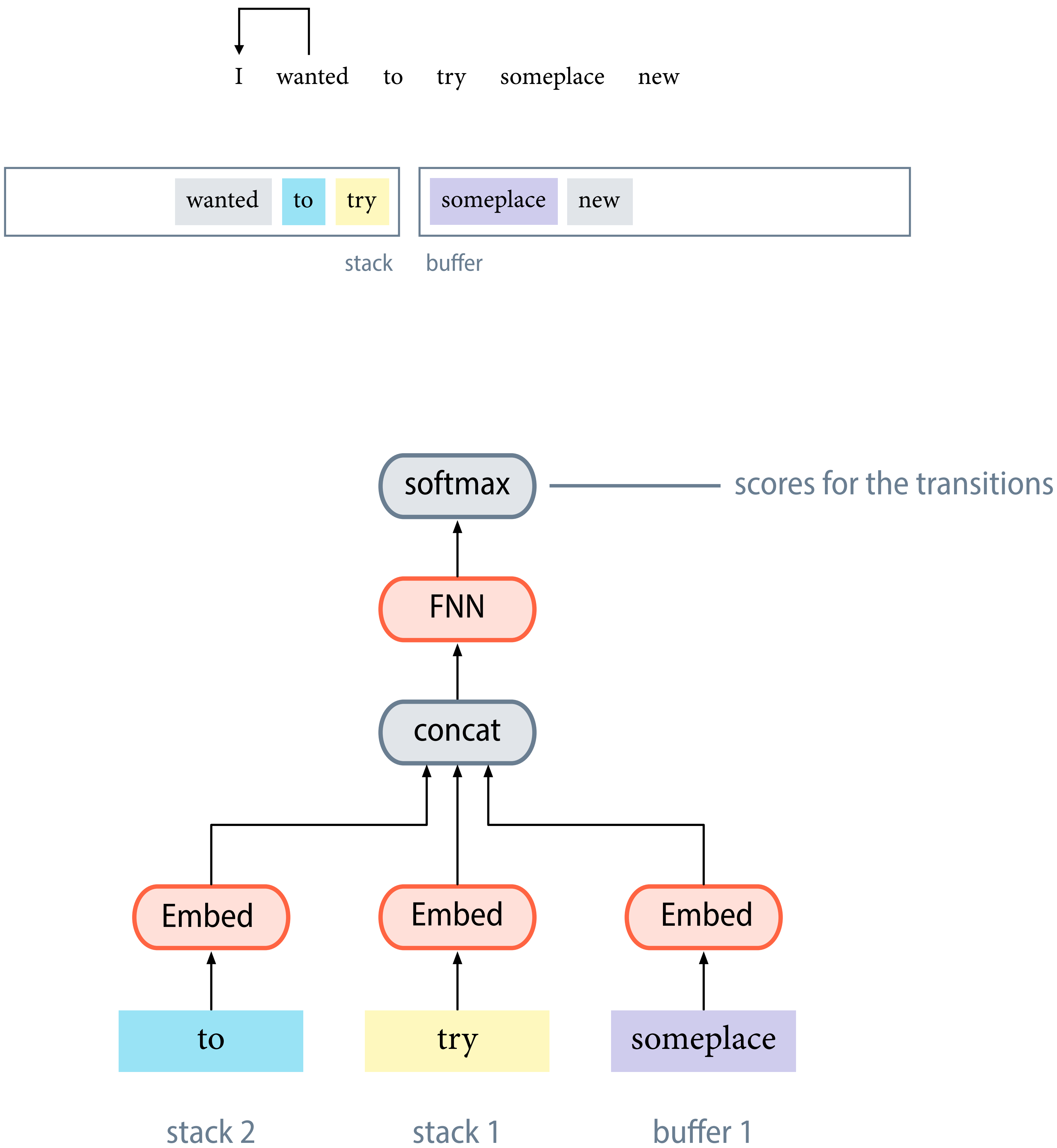
Chen and Manning (2014)

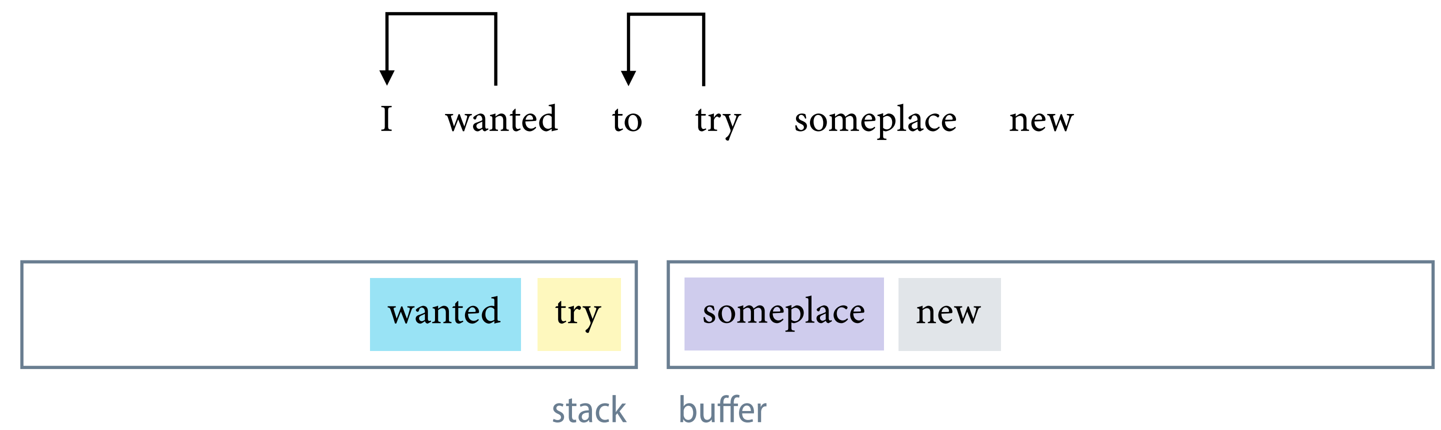
- Pre-neural transition classifiers relied on linear models with hand-crafted combination features.
- C&M propose to replace the linear model with a two-layer feedforward network (FNN).
- The standard choice for the transfer function is the rectified linear unit (ReLU).

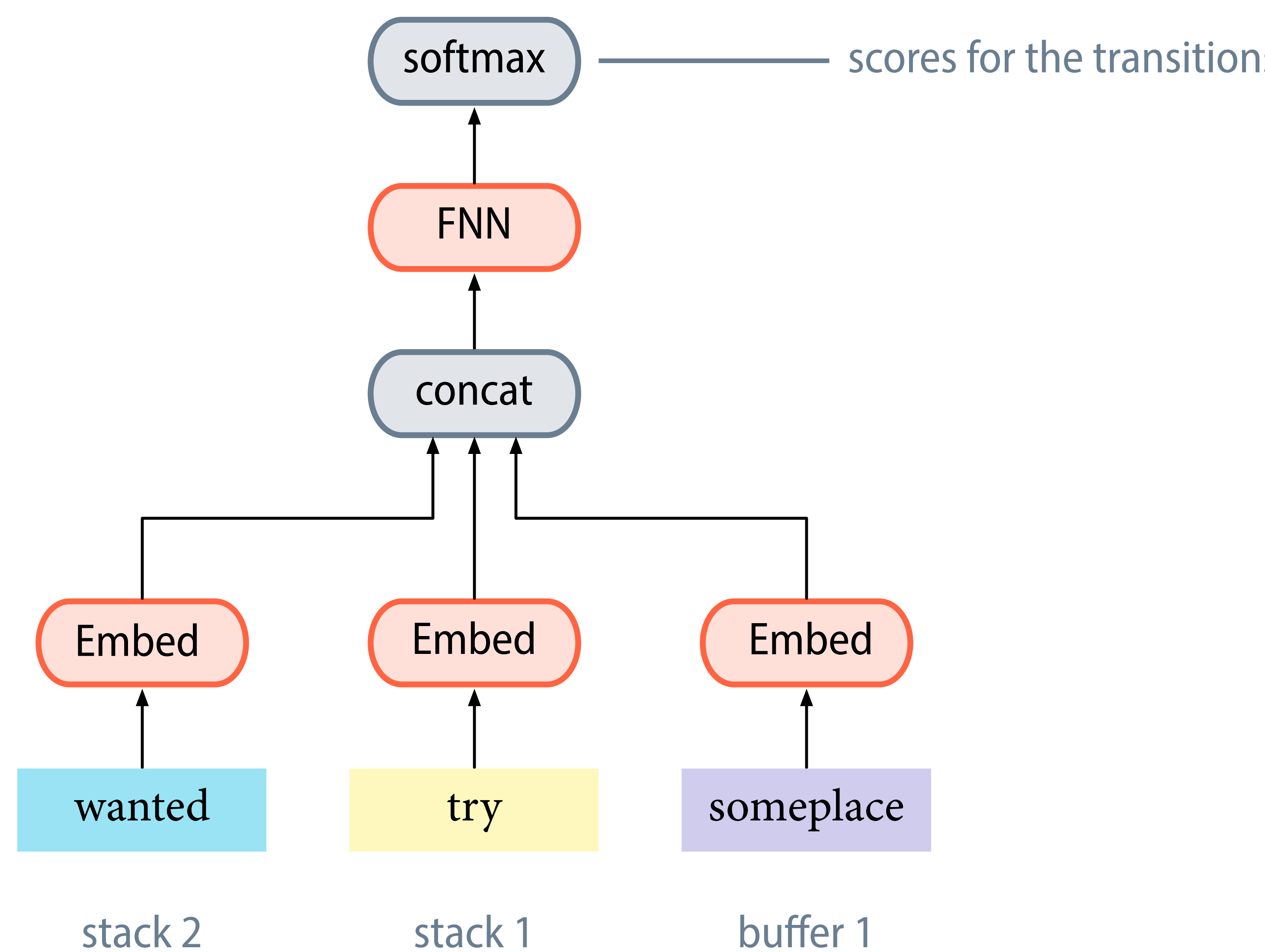
  C&M use the cube function, $f(x) = x^3$.



feedforward
neural network


I wanted to try someplace new
wanted
to
try
someplace
new
stack
buffer
softmax
scores for the transitions
FNN
concat
Embed
Embed
Embed
to
try
someplace
stack 2
stack 1
buffer 1


I wanted to try someplace new
wanted try
someplace new
stack
buffer


softmax
scores for the transitions
FNN
concat
Embed
Embed
Embed
wanted
try
someplace
stack 2
stack 1
buffer 1

Features

- C&M embed the top 3 words on the stack and buffer, as well as certain descendants of the top words on the stack.

  word embedding dimension = 50

- In addition to word embeddings, they also use embeddings for part-of-speech tags and dependency labels.

  tag embedding dimension = label embedding dimension = 50

- The resulting input dimension of the FNN is 2400.

Training

- To train their parser, C&M minimise the standard cross-entropy loss, plus an L2 regularisation term.

- To generate training examples for the transition classifier, they use the static oracle for the arc-standard algorithm.

  can be generated off-line

Parsing accuracy

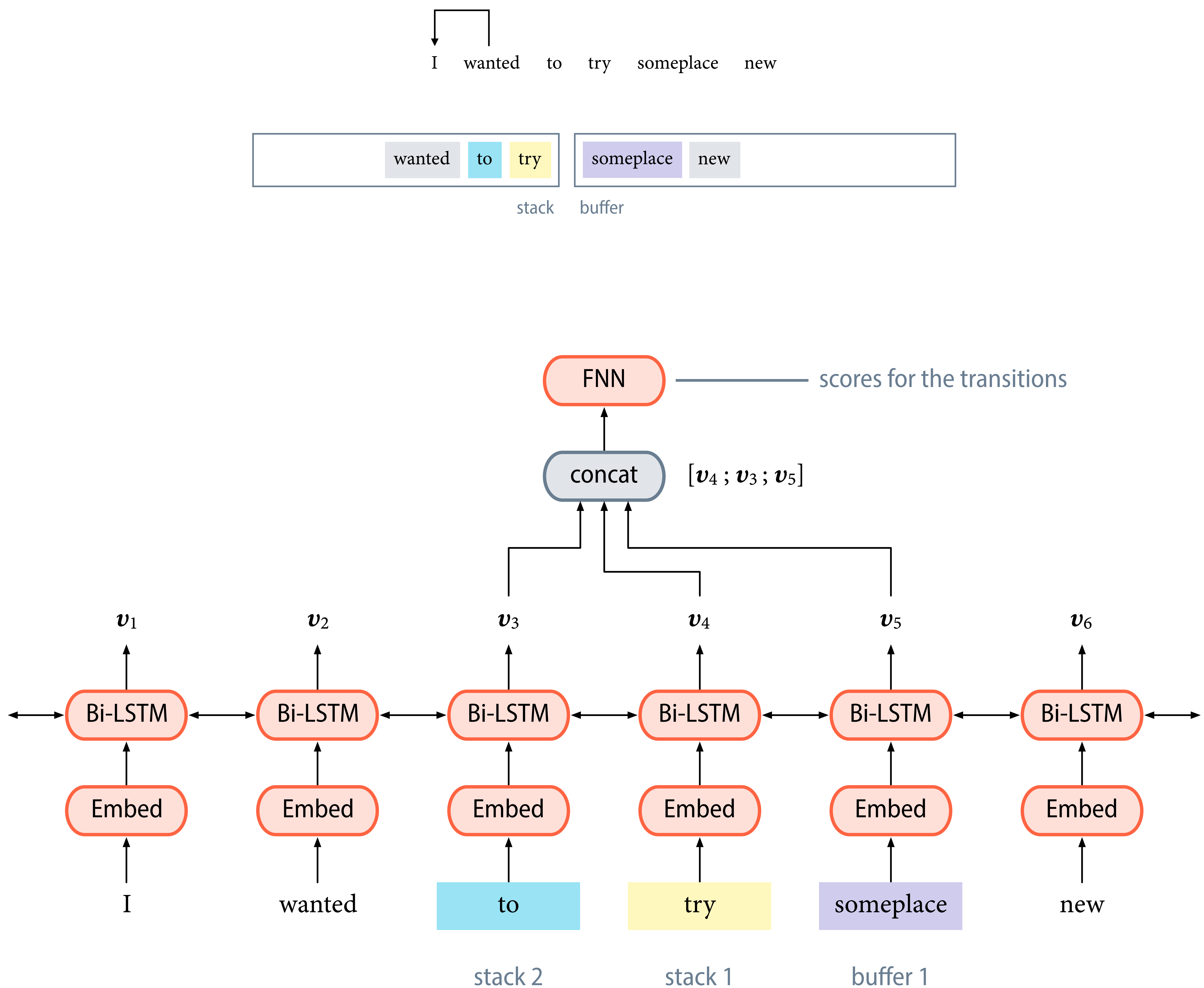
	UAS	LAS
Baseline, transition-based	89.4	87.3
Baseline, graph-based	90.7	87.6
Chen and Manning (2014)	**91.8**	**89.6**
Weiss et al. (2015)	93.2	91.2

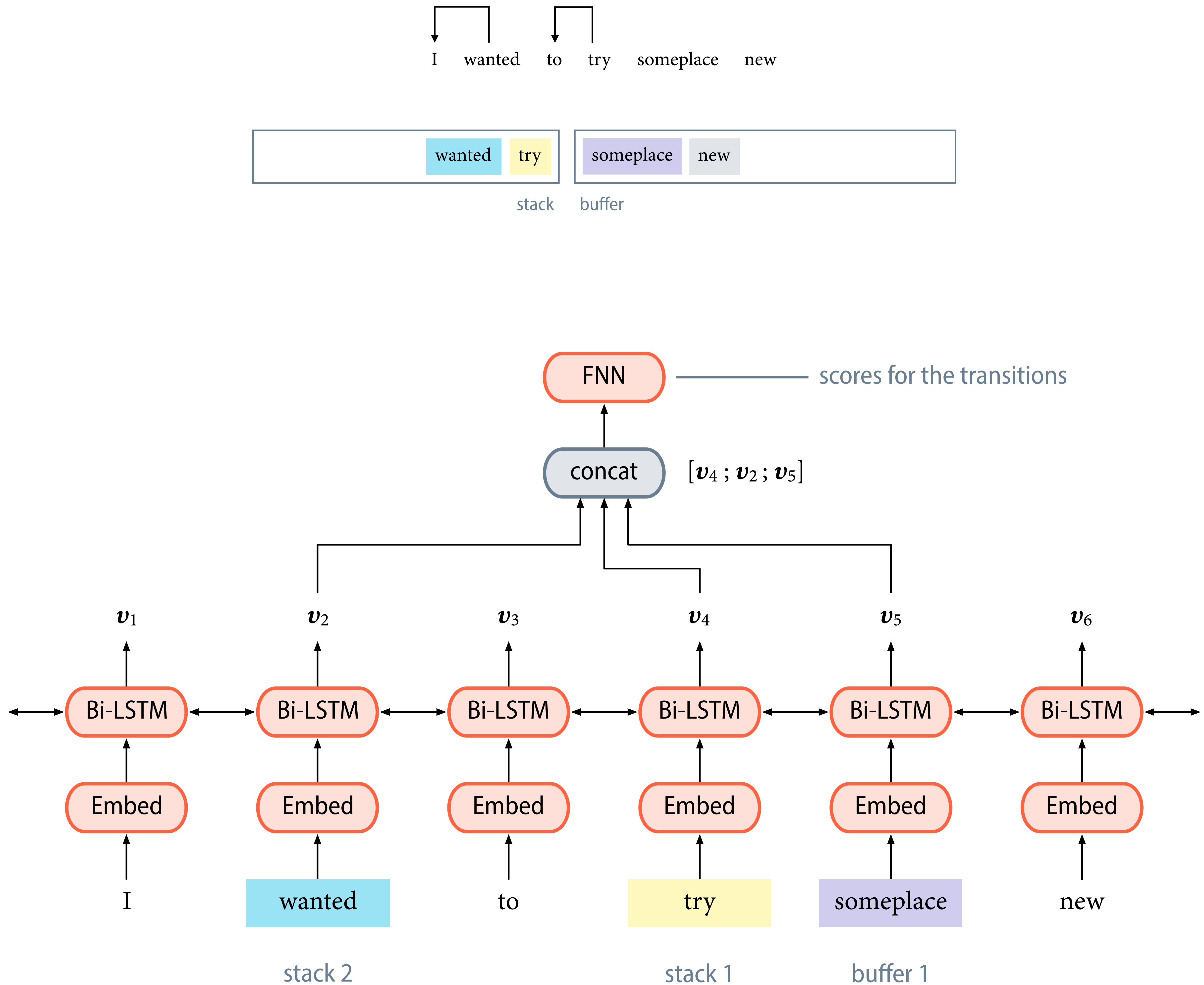
Parsing accuracy on the test set of the Penn Treebank + conversion to Stanford dependencies

Kiperwasser and Goldberg (2016)

- The neural parser of C&M learns useful feature combinations, but the need to carefully design the core features remains.
- K&G propose to use a minimal set of core features based on contextualised embeddings obtained from a Bi-LSTM.

  Bi-LSTM is trained with the rest of the parser.
- They show that this approach gives state-of-the-art accuracy both for transition-based and for graph-based parsing.


I wanted to try someplace new
wanted to try
someplace new
stack
buffer
FNN
scores for the transitions
concat
[v_4 ; v_3 ; v_5]
v_1 v_2 v_3 v_4 v_5 v_6
Bi-LSTM
Embed
I
wanted
to
try
someplace
new
stack 2
stack 1
buffer 1



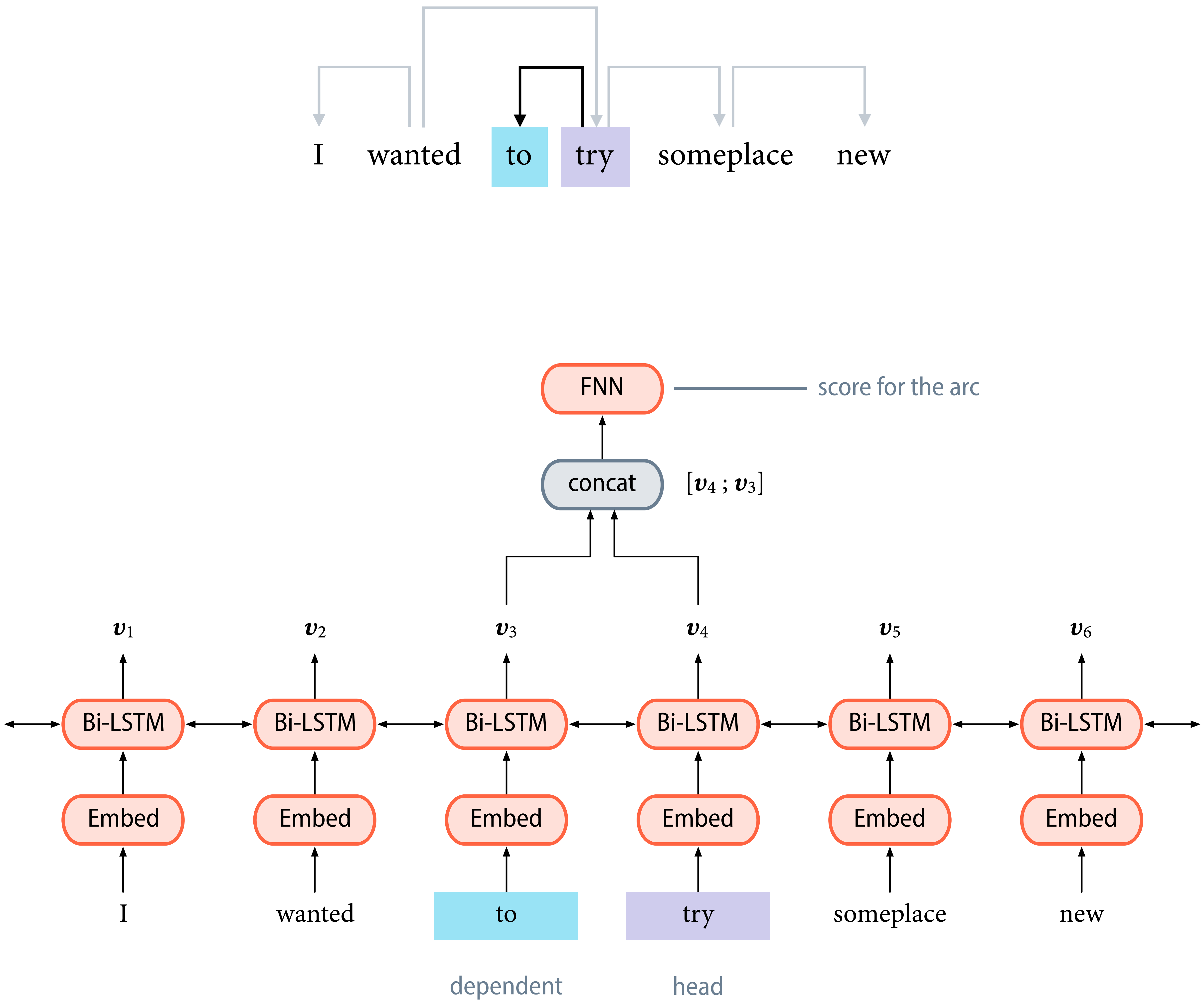
I wanted to try someplace new
wanted try
someplace new
stack
buffer
FNN
scores for the transitions
concat
[v_4 ; v_2 ; v_5]
v_1 v_2 v_3 v_4 v_5 v_6
Bi-LSTM
Embed
I wanted to try someplace new
stack 2
stack 1
buffer 1

Features and training (transition-based parser)

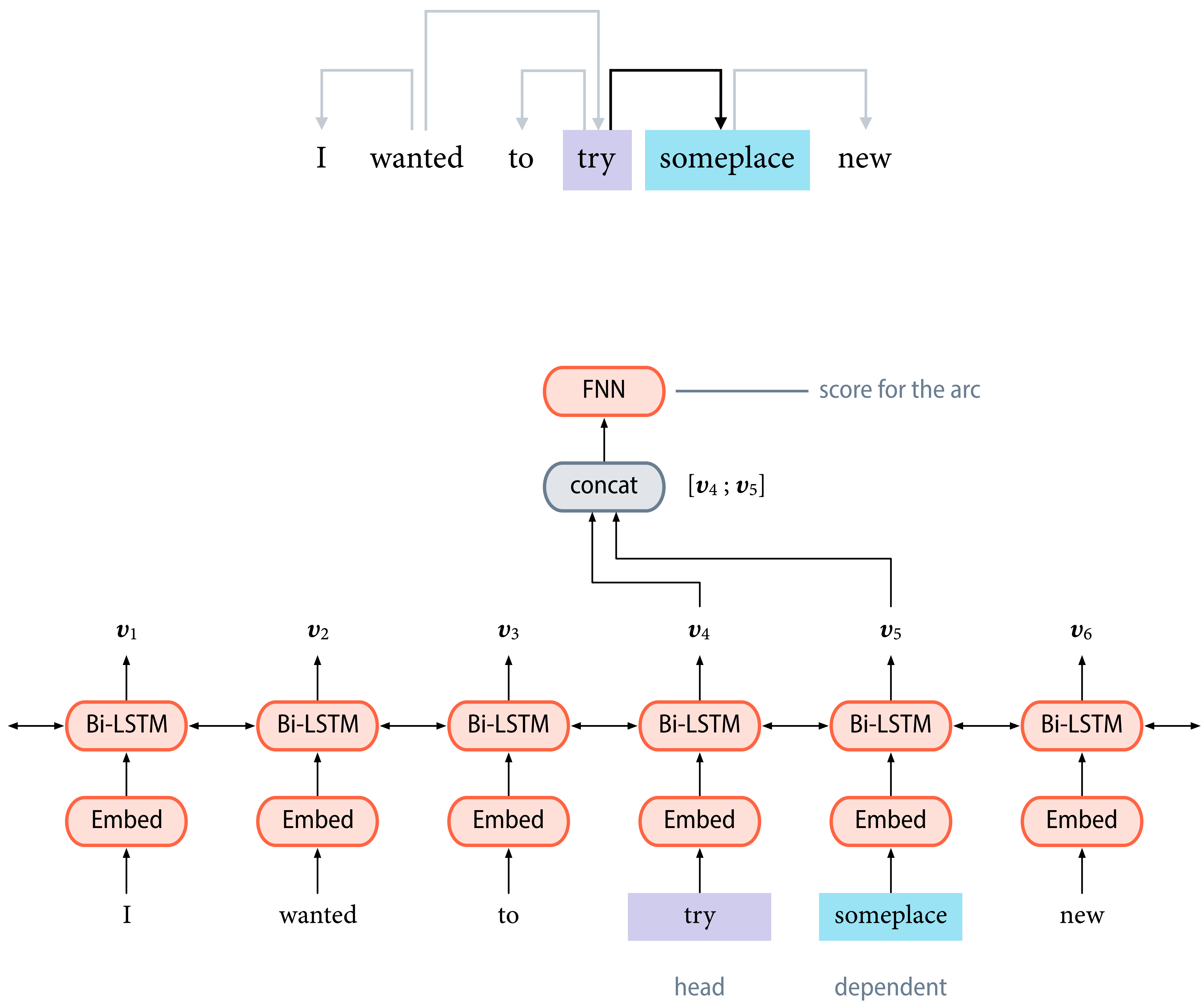
- For their transition-based parser, K&G embed the top 3 words on the stack, as well as the first word in the buffer.

  word embedding dimension = 100, tag embedding dimension = 25

- In contrast to C&M, they use a dynamic oracle, so they cannot generate training examples in an off-line fashion.



I
wanted
to
try
someplace
new
FNN
score for the arc
concat
$[\boldsymbol{v}_4 ; \boldsymbol{v}_3]$
$\boldsymbol{v}_1$
$\boldsymbol{v}_2$
$\boldsymbol{v}_3$
$\boldsymbol{v}_4$
$\boldsymbol{v}_5$
$\boldsymbol{v}_6$
Bi-LSTM
Bi-LSTM
Bi-LSTM
Bi-LSTM
Bi-LSTM
Bi-LSTM
Embed
Embed
Embed
Embed
Embed
Embed
I
wanted
to
try
someplace
new
dependent
head



I
wanted
to
try
someplace
new
FNN
score for the arc
concat
[v_4 ; v_5]
v_1
v_2
v_3
v_4
v_5
v_6
Bi-LSTM
Embed
I
wanted
to
try
someplace
new
head
dependent

Features and training (graph-based parser)

- For their graph-based parser, K&G embed the head and dependent of each arc.

  word embedding dimension = 100, tag embedding dimension = 25

- The training objective is to maximise the margin between the score of the gold tree y^* and the highest-scoring incorrect tree y:

$$L(\boldsymbol{\theta}) = \max\left(0, 1 + \max_{y \neq y^*} \operatorname{score}(x, y) - \operatorname{score}(x, y^*)\right)$$

Parsing accuracy

	UAS	LAS
Chen and Manning (2014)	91.8	89.6
Weiss et al. (2015)	93.2	91.2
K & G (2016), graph-based	93.0	90.9
K & G (2016), transition-based	**93.6**	**91.5**

Parsing accuracy on the test set of the Penn Treebank + conversion to Stanford dependencies

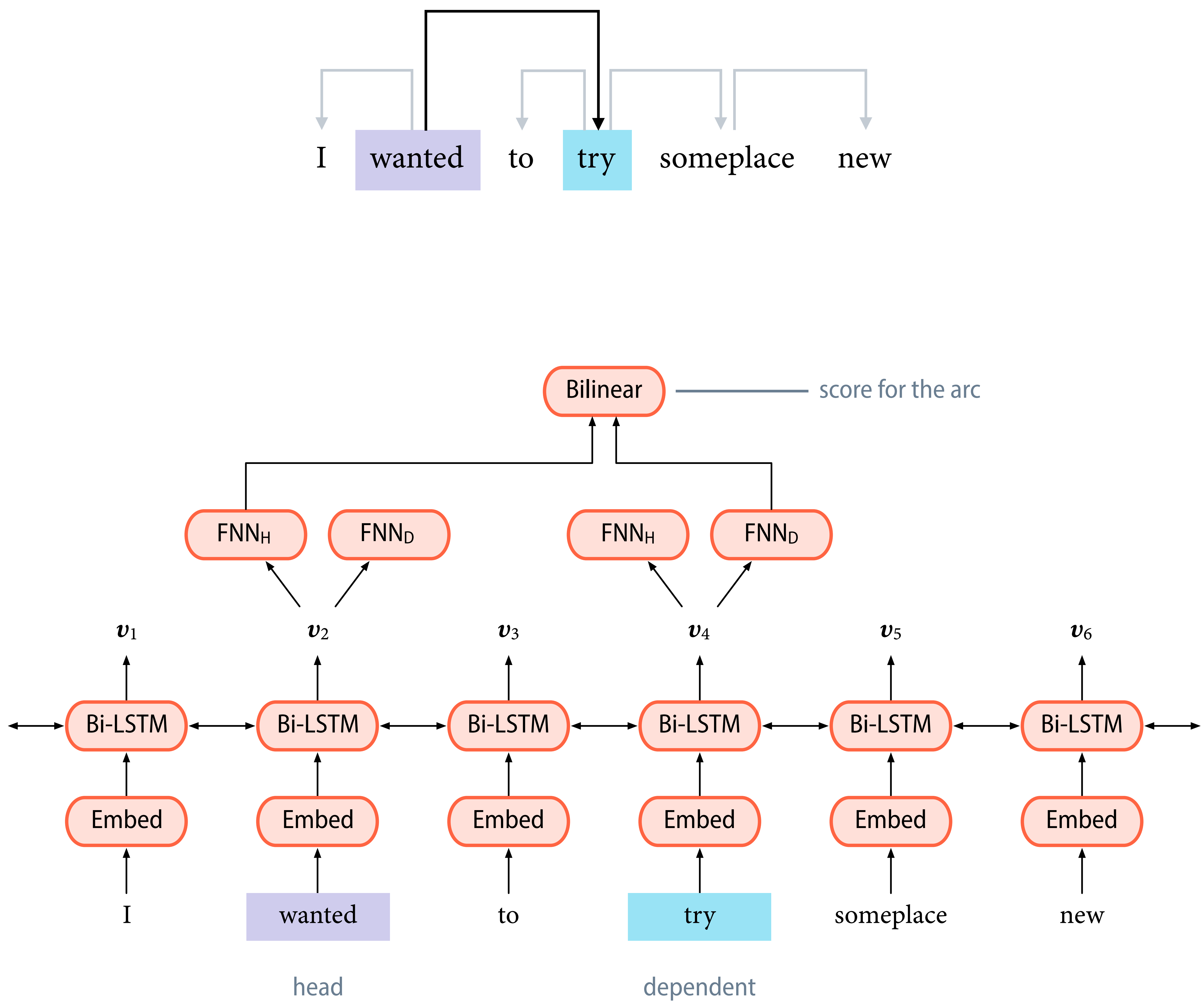
Dozat and Manning (2017)

- Based on the context-dependent embeddings, two FNNs create specialised representations of each word as a head/dependent:

$$\boldsymbol{h}_i = \mathrm{FNN}_h(\boldsymbol{v}_i) \qquad \boldsymbol{d}_i = \mathrm{FNN}_d(\boldsymbol{v}_i)$$

- These specialised representations are then scored via a bilinear layer with weight tensor $\boldsymbol{U}$ and bias vector $\boldsymbol{b}$:

$$\mathrm{score}(x, i \to j) = \boldsymbol{h}_i \, \boldsymbol{U} \, \boldsymbol{d}_j^\top + (\boldsymbol{h}_i \, \boldsymbol{b})^\top$$


I
wanted
to
try
someplace
new
Bilinear
score for the arc
FNNH
FNND
FNNH
FNND
v_1
v_2
v_3
v_4
v_5
v_6
Bi-LSTM
Bi-LSTM
Bi-LSTM
Bi-LSTM
Bi-LSTM
Bi-LSTM
Embed
Embed
Embed
Embed
Embed
Embed
I
wanted
to
try
someplace
new
head
dependent

Parsing accuracy

	UAS	LAS
Chen and Manning (2014)	91.8	89.6
K & G (2016), graph-based	93.0	90.9
K & G (2016), transition-based	93.6	91.5
Dozat and Manning (2017)	**95.7**	**94.1**

Parsing accuracy on the test set of the Penn Treebank + conversion to Stanford dependencies