Neural architectures for dependency parsing

Marco Kuhlmann
Department of Computer and Information Science

LINKÖPING UNIVERSITY

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)

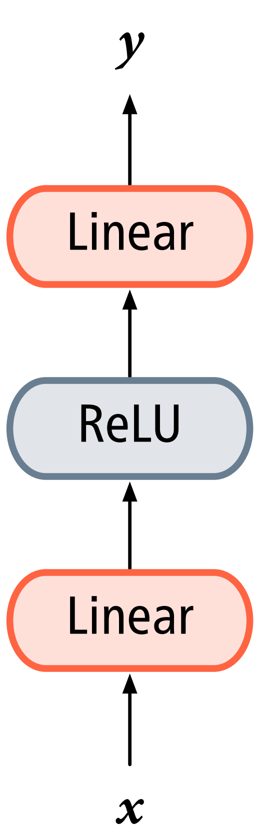
- Learning an arc-factored graph-based dependency parser amounts to learning the arc scores.

  Kiperwasser and Goldberg (2016), Glavaš and Vulić (2021)

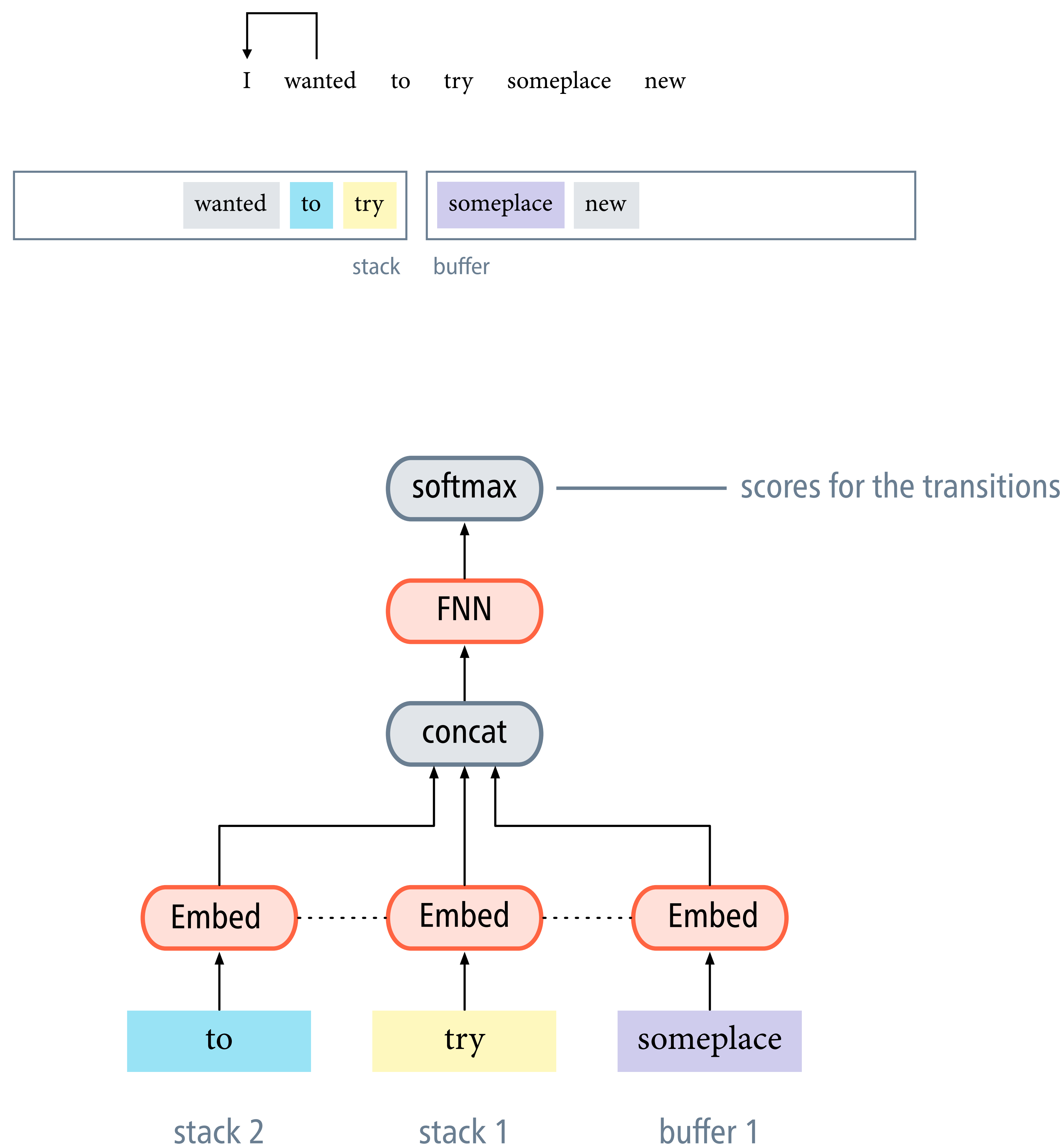
Chen and Manning (2014)

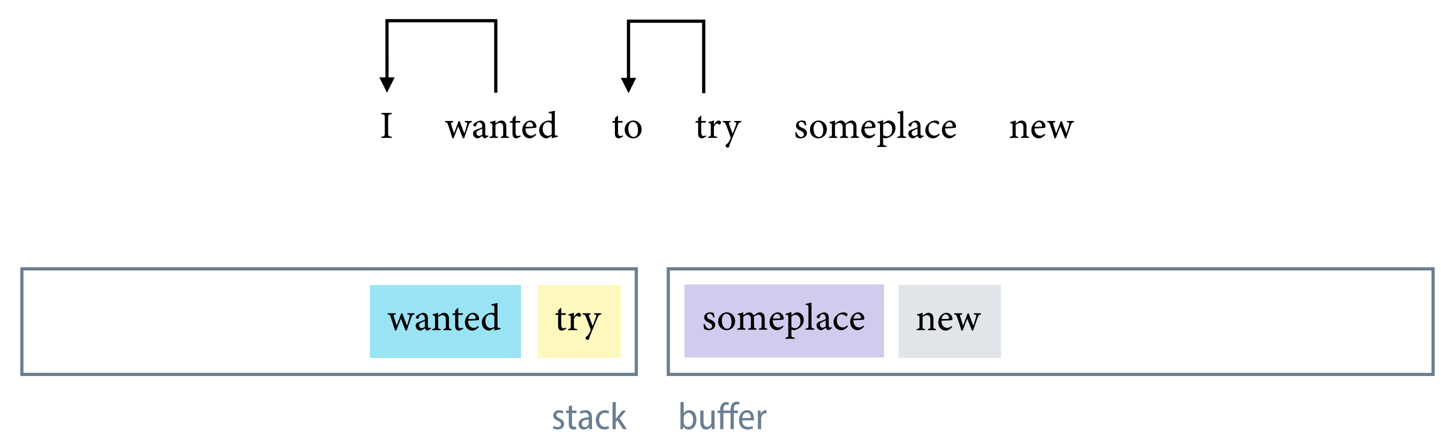
- Pre-neural transition classifiers relied on linear models with hand-crafted combination features.
- C & M propose to replace the linear model with a two-layer feedforward network (FNN).
- The standard choice for the transfer function is the rectified linear unit (ReLU).

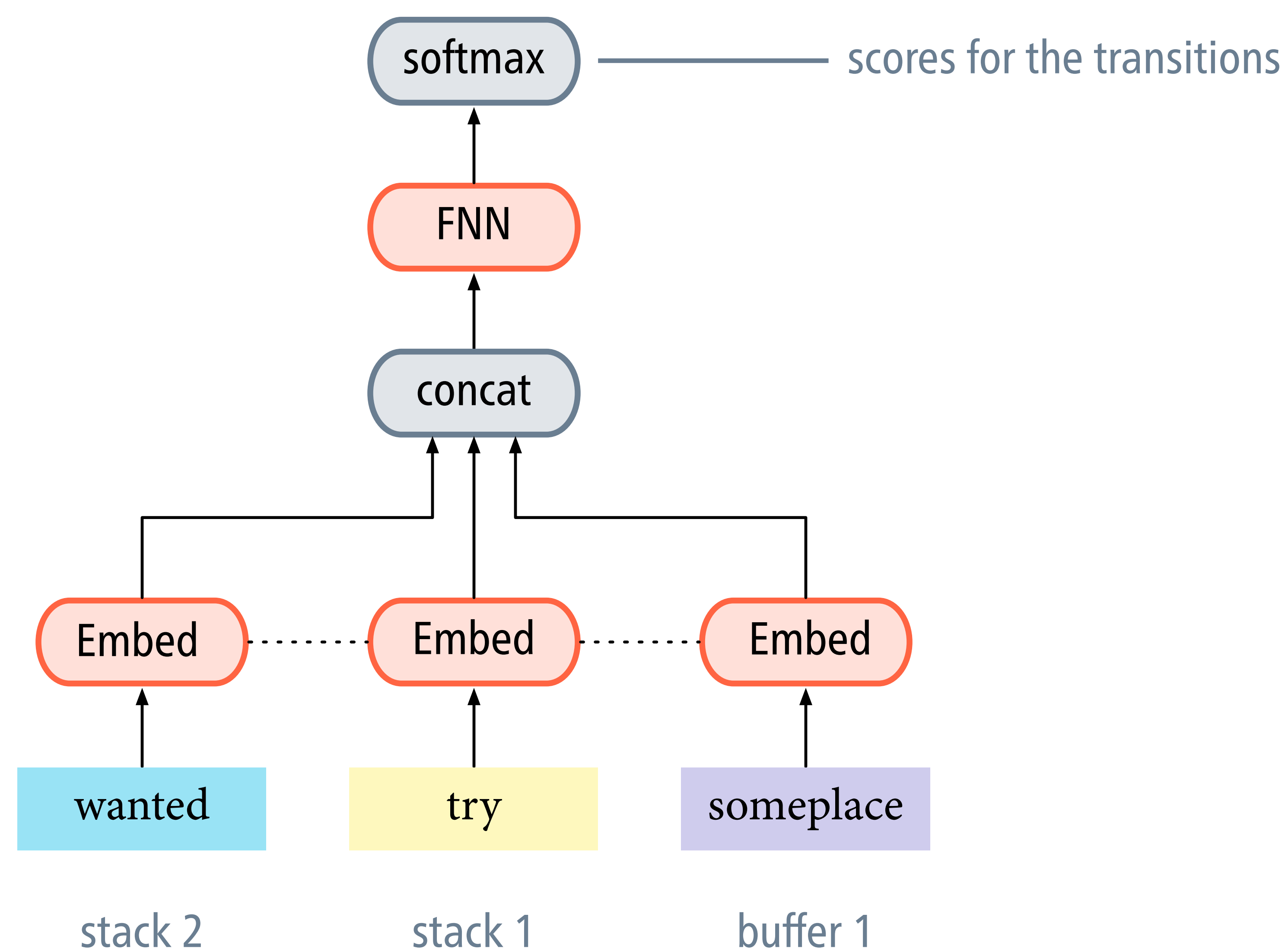
  C & M use the cube function, $f(x) = x^3$.



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

Chen and Manning (2014) – 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.
- In addition to word embeddings, they also use embeddings for part-of-speech tags and dependency labels.

Chen and Manning (2014) – Training

- To train their parser, C & M minimise cross-entropy loss relative to the gold-standard action, 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

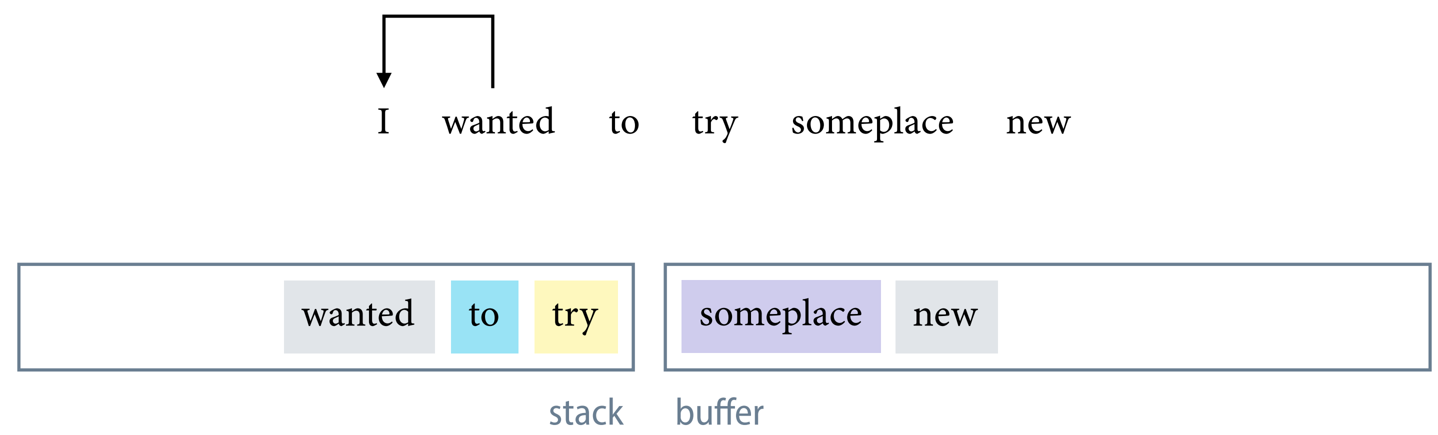
	UAS	LAS
Baseline, transition-based	89.4	87.3
Baseline, graph-based	90.7	87.6
Chen and Manning (2014)	**91.8**	**89.6**
Weiss et al. (2015)	93.2	91.2

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

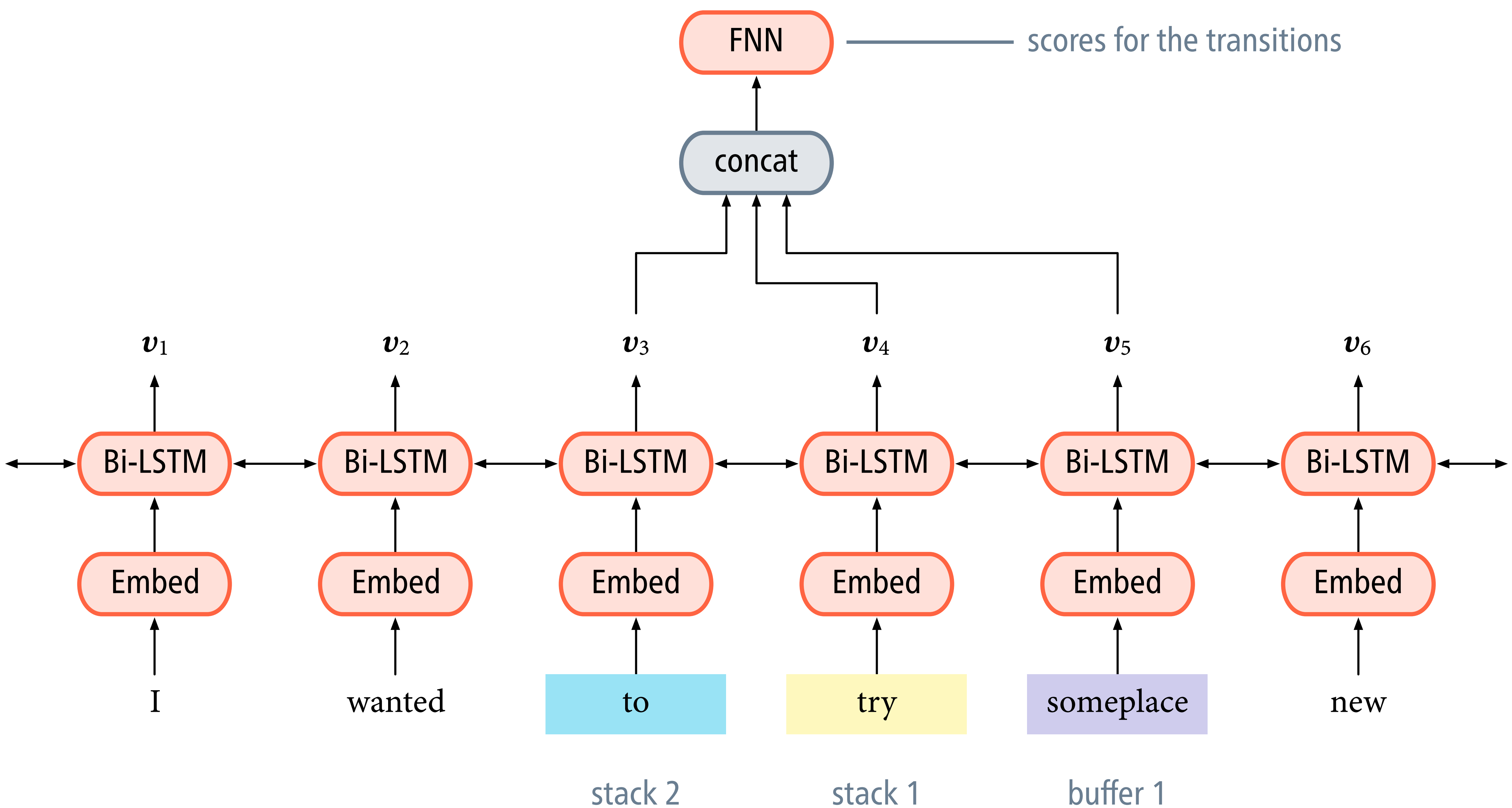
Kiperwasser and Goldberg (2016)

- 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.

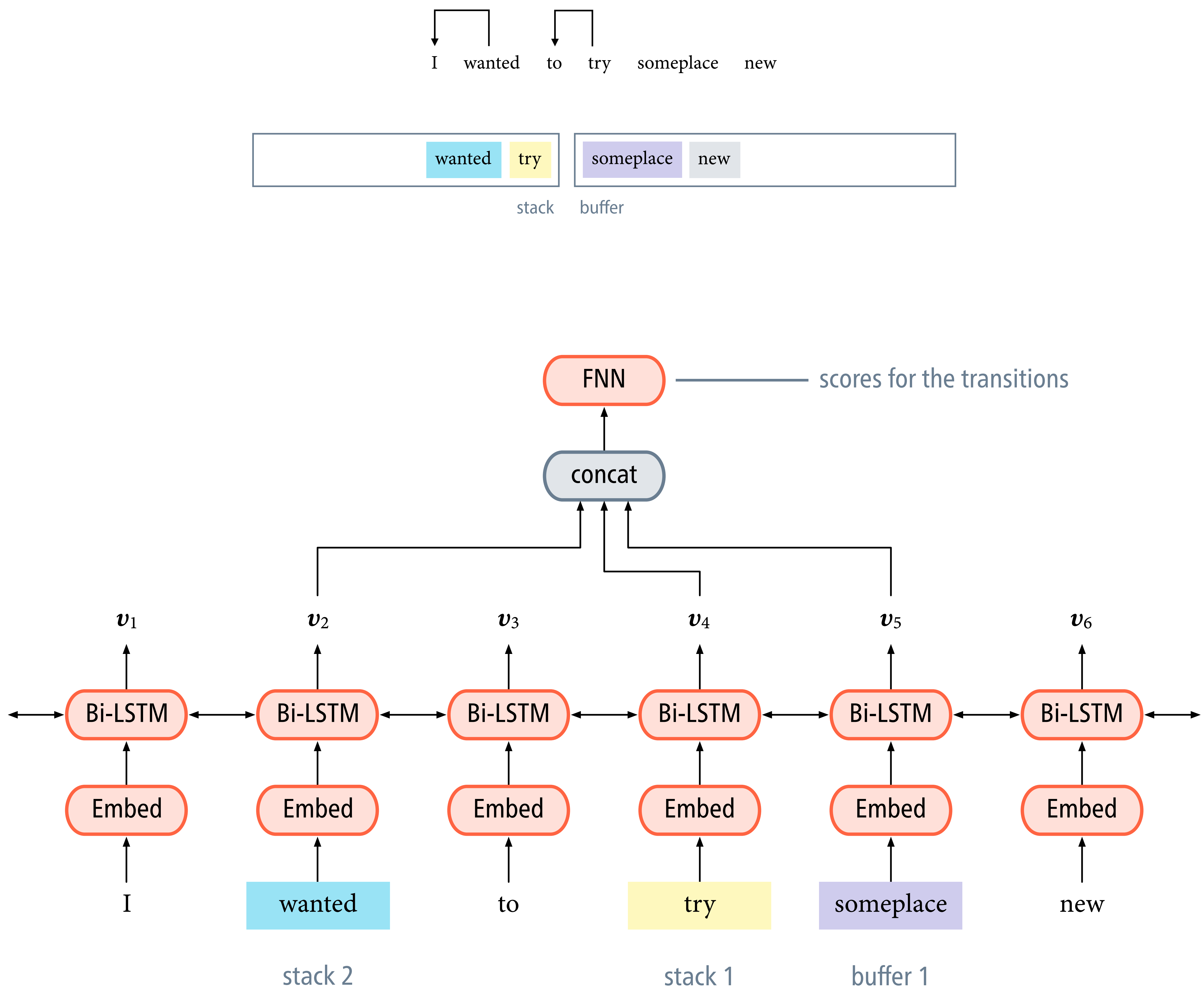
  Bi-LSTM is trained with the rest of the parser.
- They show that this approach gives state-of-the-art accuracy both for transition-based and for graph-based parsing.



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



FNN
scores for the transitions
concat
v_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

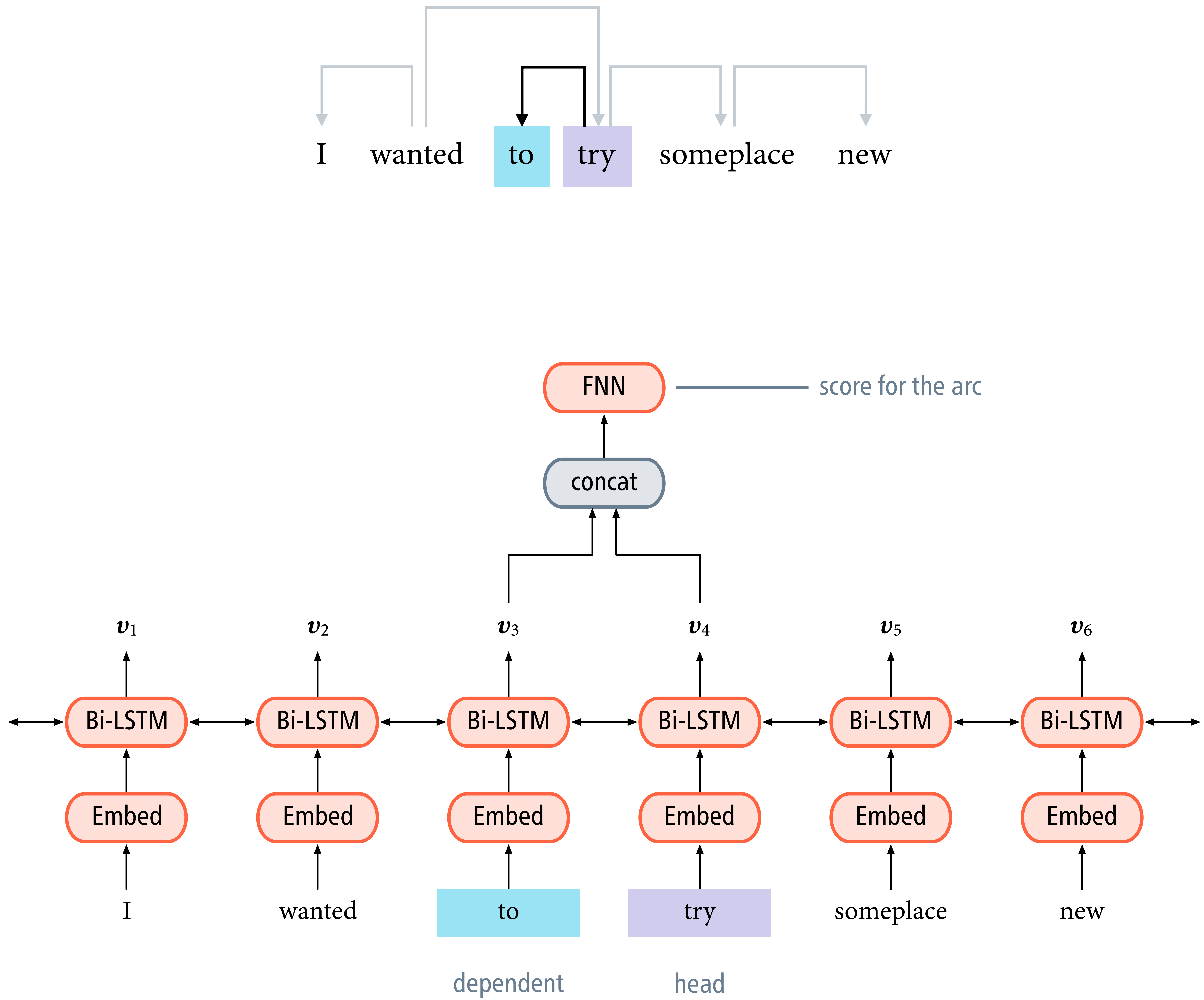

I
wanted
to
try
someplace
new
wanted
try
someplace
new
stack
buffer
FNN
scores for the transitions
concat
v_1
v_2
v_3
v_4
v_5
v_6
Bi-LSTM
Embed
I
wanted
to
try
someplace
new
stack 2
stack 1
buffer 1

Features and training (transition-based parser)

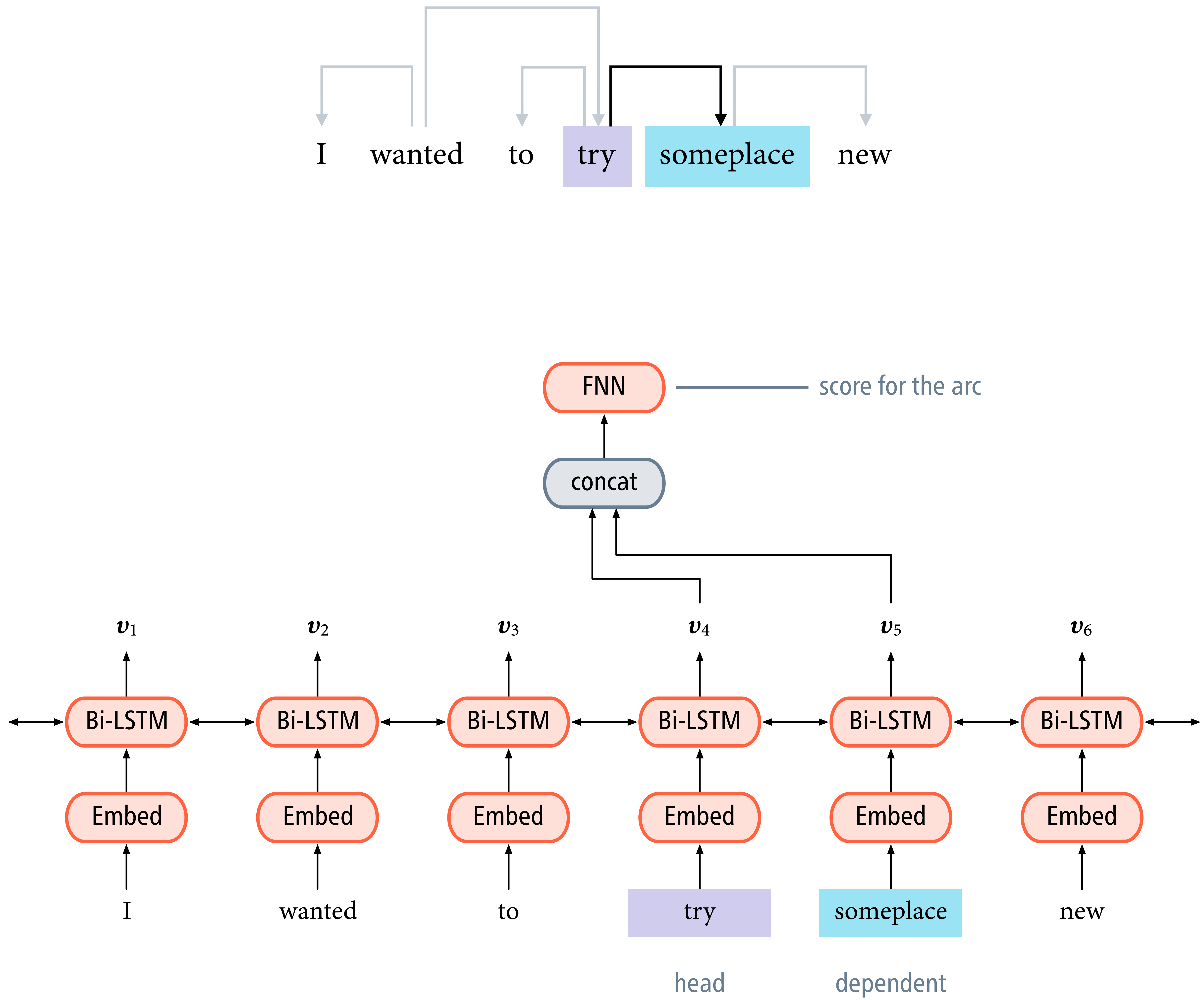
- For their transition-based parser, K & G embed the top 3 words on the stack, as well as the first word in the buffer.

  both word and part-of-speech tag

- 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
v_1 v_2 v_3 v_4 v_5 v_6
Bi-LSTM
Embed
I wanted to try someplace new
dependent
head


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

Features and training (graph-based parser)

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

  both word and part-of-speech tag

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

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

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

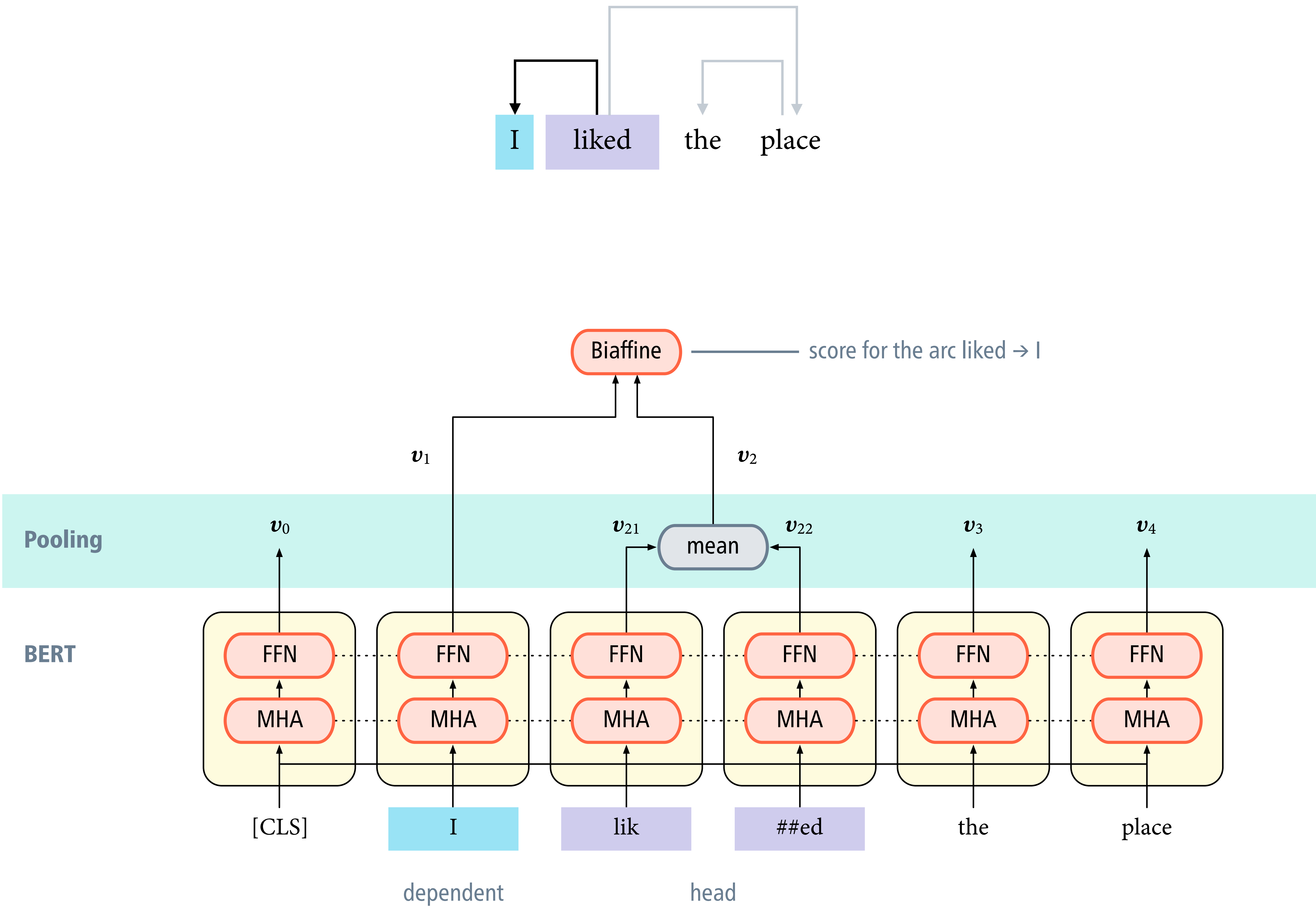
Glavaš and Vulić (2021)

- G & V adopt the basic architecture of K & G but use a BERT encoder instead of a Bi-LSTM.

  requires word-level average pooling of token representations

- The arc scores are computed using a bi-affine layer:

$$\mathrm{score}(x, i \rightarrow j) = \boldsymbol{w}_i \boldsymbol{W} \boldsymbol{w}_j^\top + \boldsymbol{b}$$



I
liked
the
place
Biaffine
score for the arc liked → I
v_1
v_2
Pooling
v_0
v_{21}
mean
v_{22}
v_3
v_4
BERT
FFN
MHA
[CLS]
I
lik
##ed
the
place
dependent
head