Carnegie Mellon University
Language Technologies Institute

Neuro-Symbolic Language Modeling with Retrieval Automaton

Uri Alon
Language Technologies Institute
Carnegie Mellon University

Frank F. Xu
CMU

Junxian He
CMU

Sudipta Sengupta
Amazon AWS

Dan Roth
AWS AI Labs

Graham Neubig
CMU

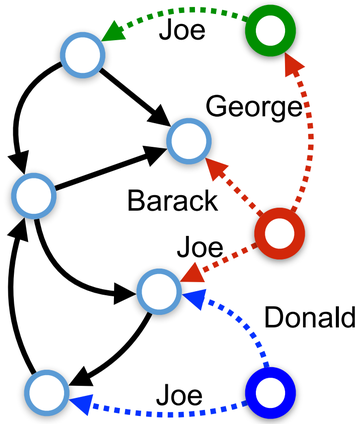
RetoMaton - TL;DR:

Given a trained LM and its training corpus, we construct a **weighted finite-state automaton**.

At inference time, we traverse the automaton in parallel with the LM.

We interpolate this automaton's probability with the base LM's probability.

Automaton

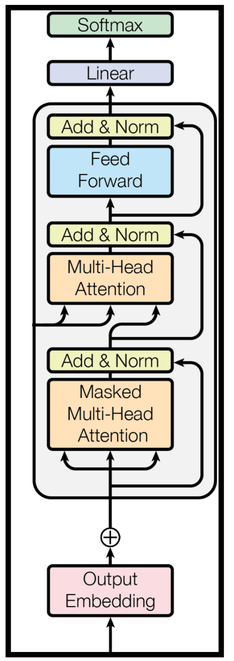


States: clusters of training examples, encoded by the LM

Edges: pointers between consecutive examples, shared in cluster

Weights: $-\|h^{(t)}, h_i\|_2$

Trained LM



$$\lambda P_{auto} + (1 - \lambda) P_{LM}$$

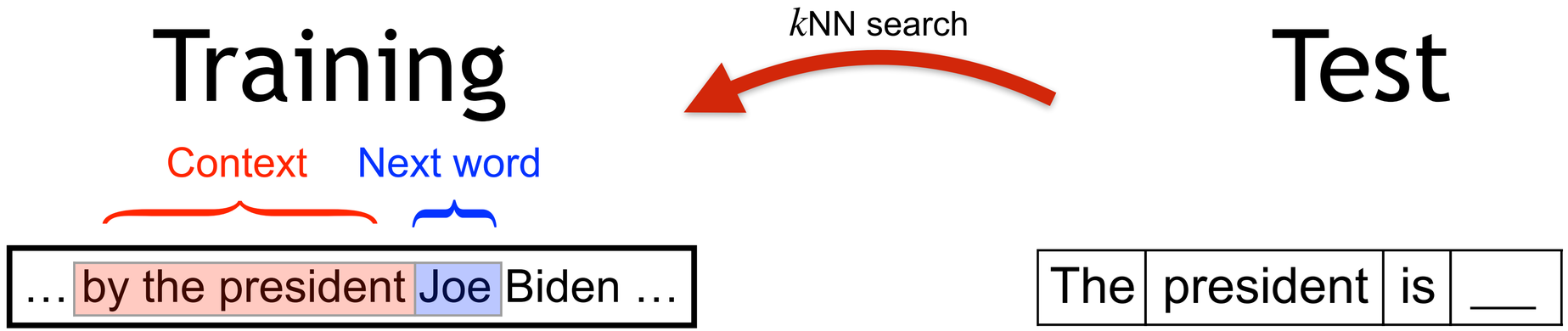
Background: K-Nearest Neighbor Language Model (kNN-LM)

(Khandelwal et al., ICLR'2020)

The	president	is	___

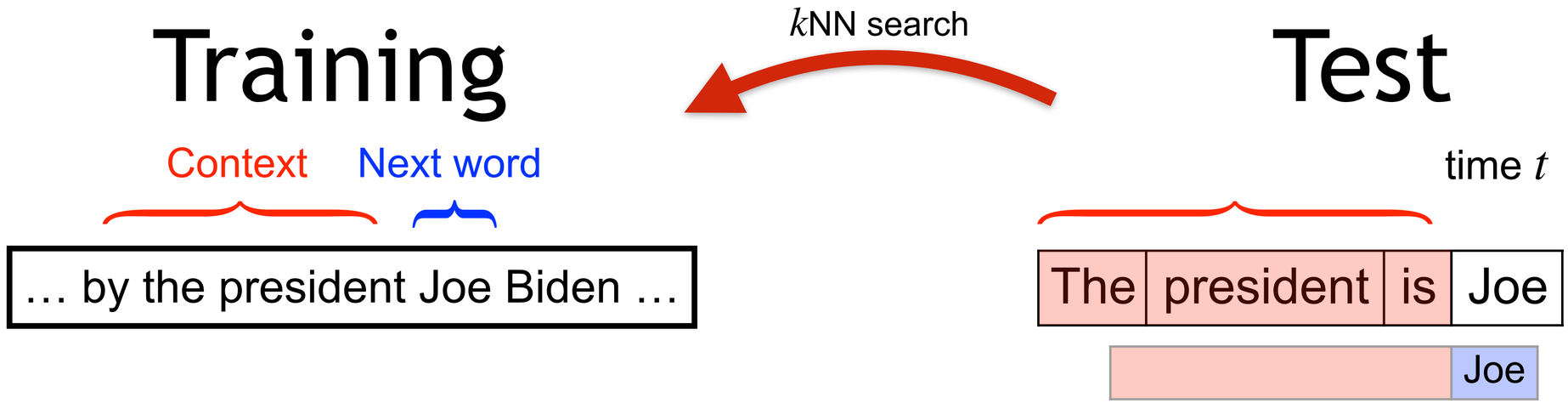
Background: K-Nearest Neighbor Language Model (kNN-LM)

(Khandelwal et al., ICLR'2020)



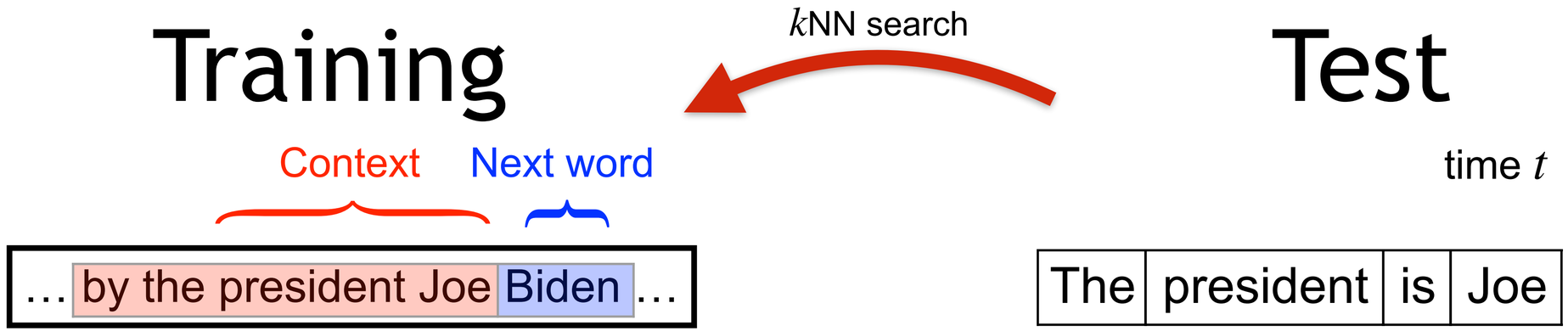
Background: K-Nearest Neighbor Language Model (kNN-LM)

(Khandelwal et al., ICLR'2020)



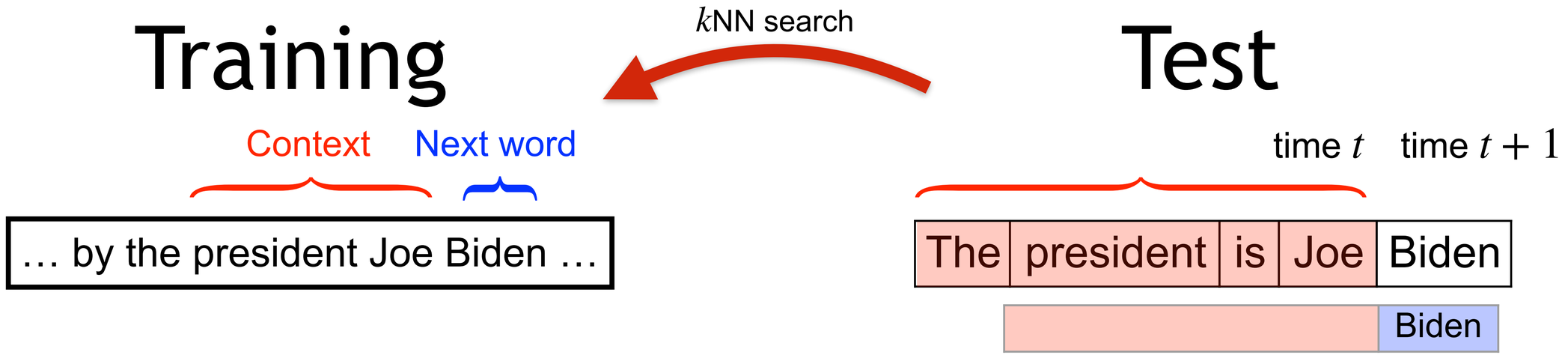
Background: K-Nearest Neighbor Language Model (kNN-LM)

(Khandelwal et al., ICLR'2020)

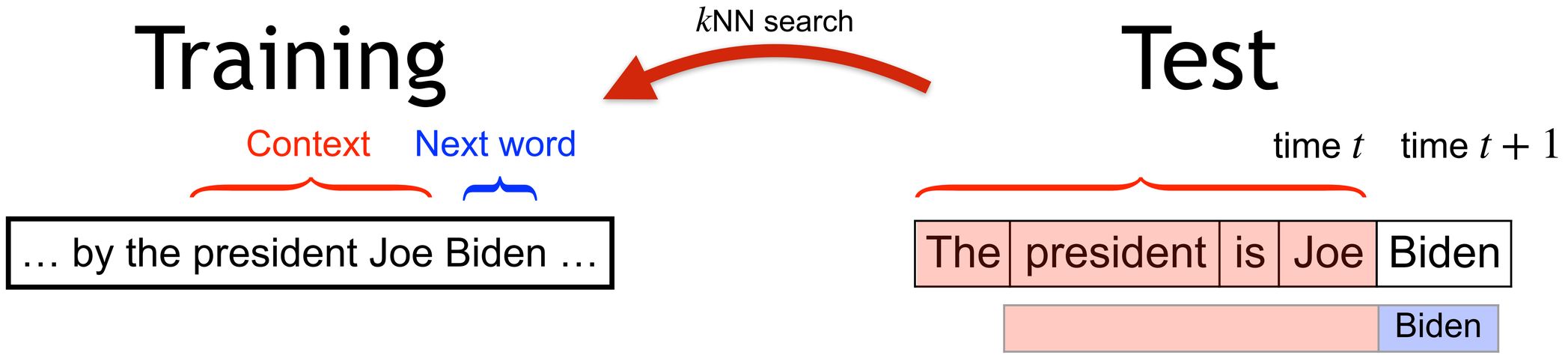


Background: K-Nearest Neighbor Language Model (kNN-LM)

(Khandelwal et al., ICLR'2020)



Background: K-Nearest Neighbor Language Model (kNN-LM)
(Khandelwal et al., ICLR'2020)



K-nearest neighbor search: for **every generated token**

$$\text{time}\left(k\text{NN search}\right) >> \text{time}\left(\text{forward pass}\right)$$

If we performed **kNN search** to retrieve "Joe",
can we save the search when predicting "Biden"?

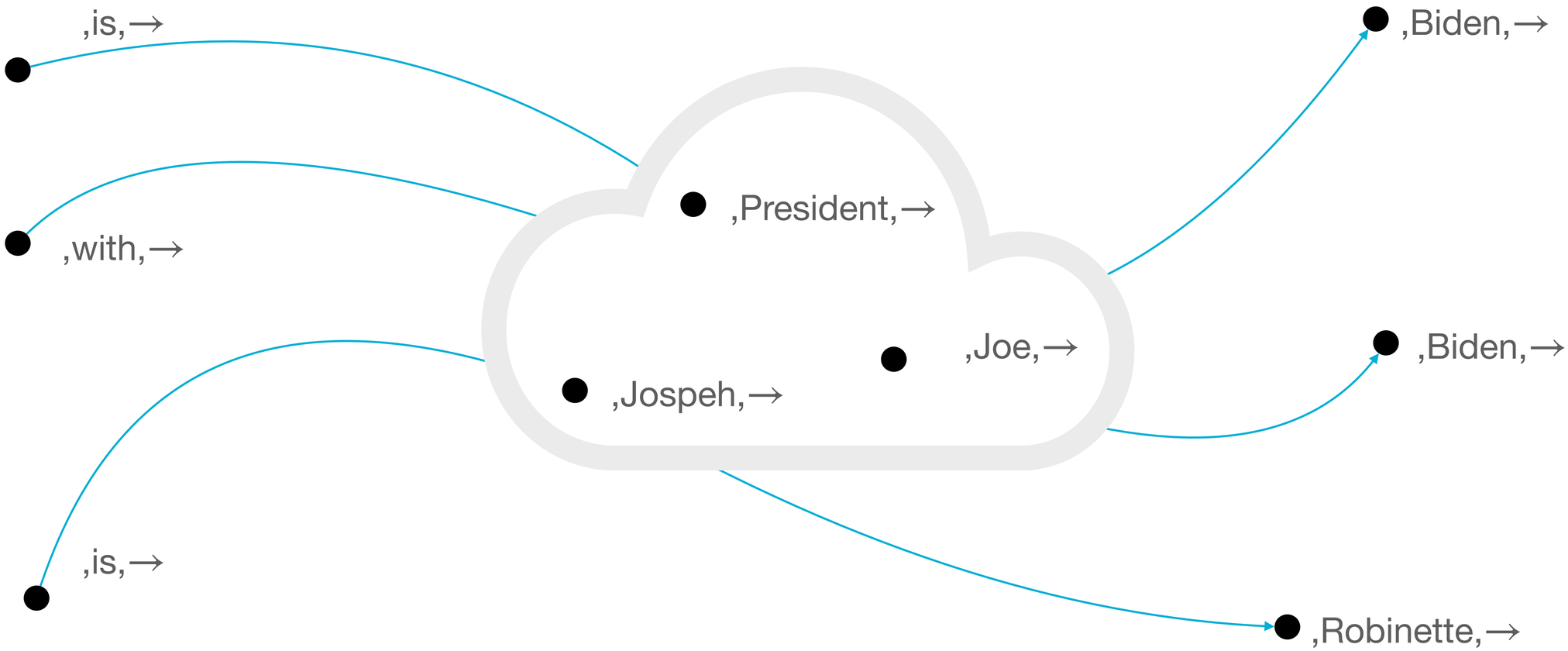
Adding Pointers Between Datastore Entries

Training

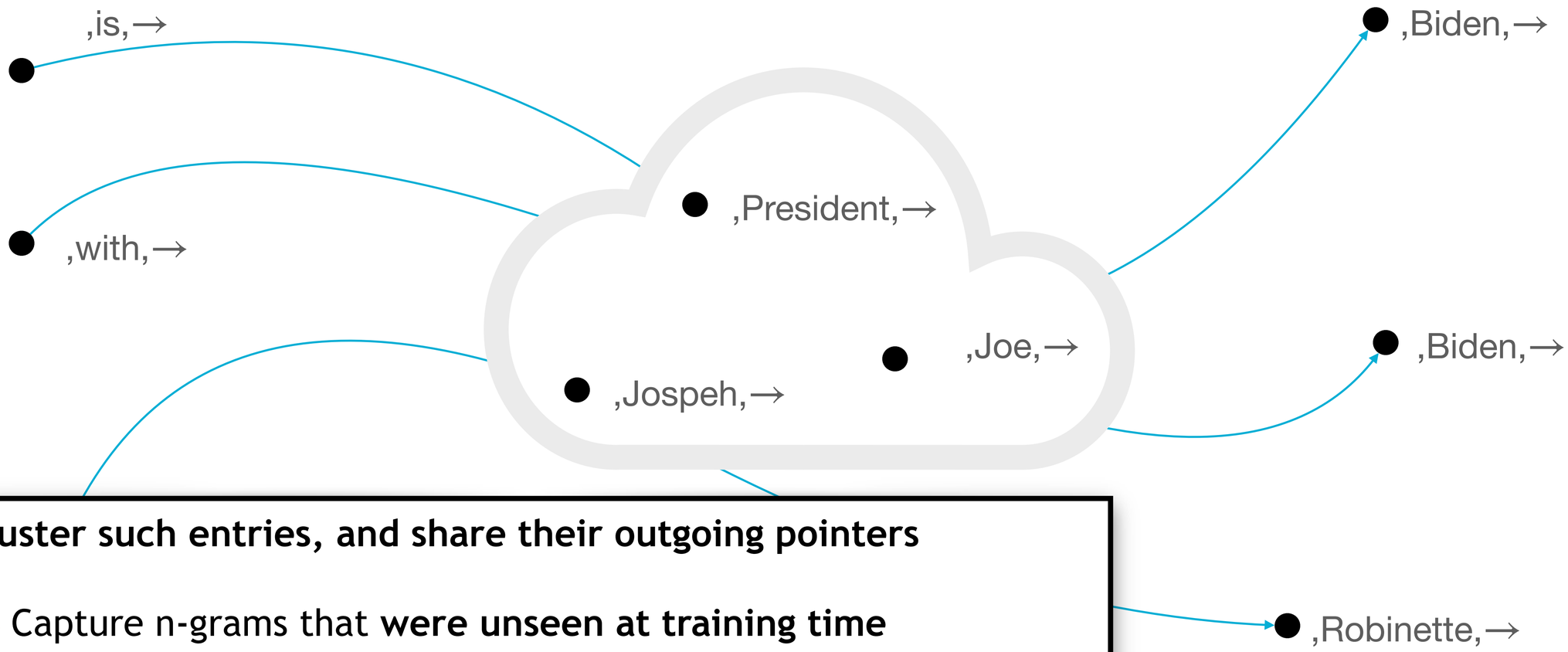
… by the president Joe Biden …

We still need to perform k**NN search** once, but in the following time steps, we can just follow pointers instead!

Clustering Entries with Close Keys



Clustering Entries with Close Keys



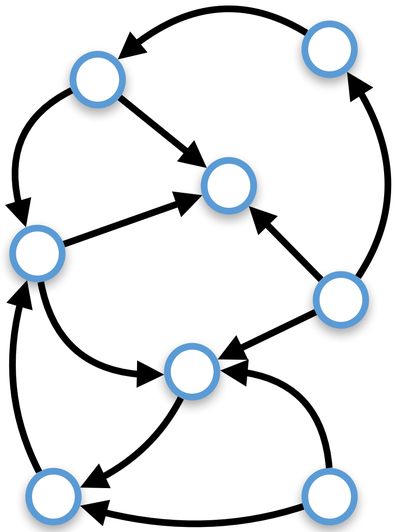
Cluster such entries, and share their outgoing pointers

👍 Capture n-grams that **were unseen at training time**

👍 Longer pointer traversal, without backing up to kNN search

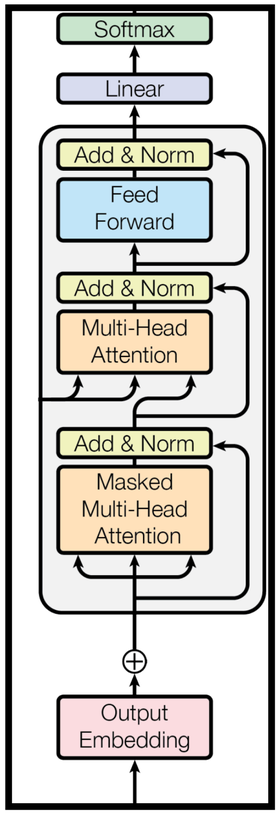
RetoMaton

Automaton



$$\lambda P_{auto} + (1 - \lambda)\ P_{LM}$$

Trained LM



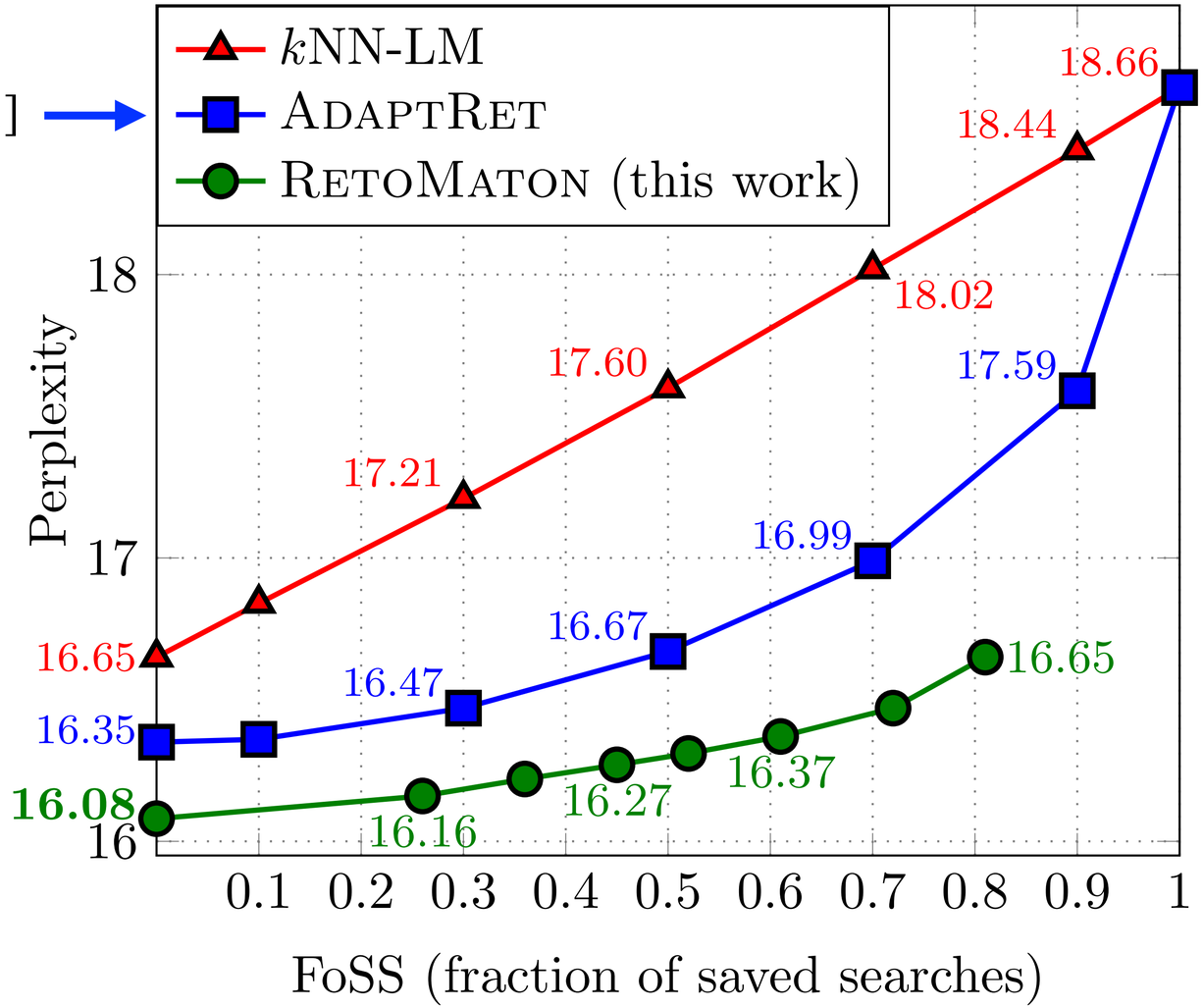
States: clusters of training examples, encoded by the LM

Edges: pointers between consecutive examples, shared in cluster

Weights: $-\|h^{(t)}, h_i\|_2$

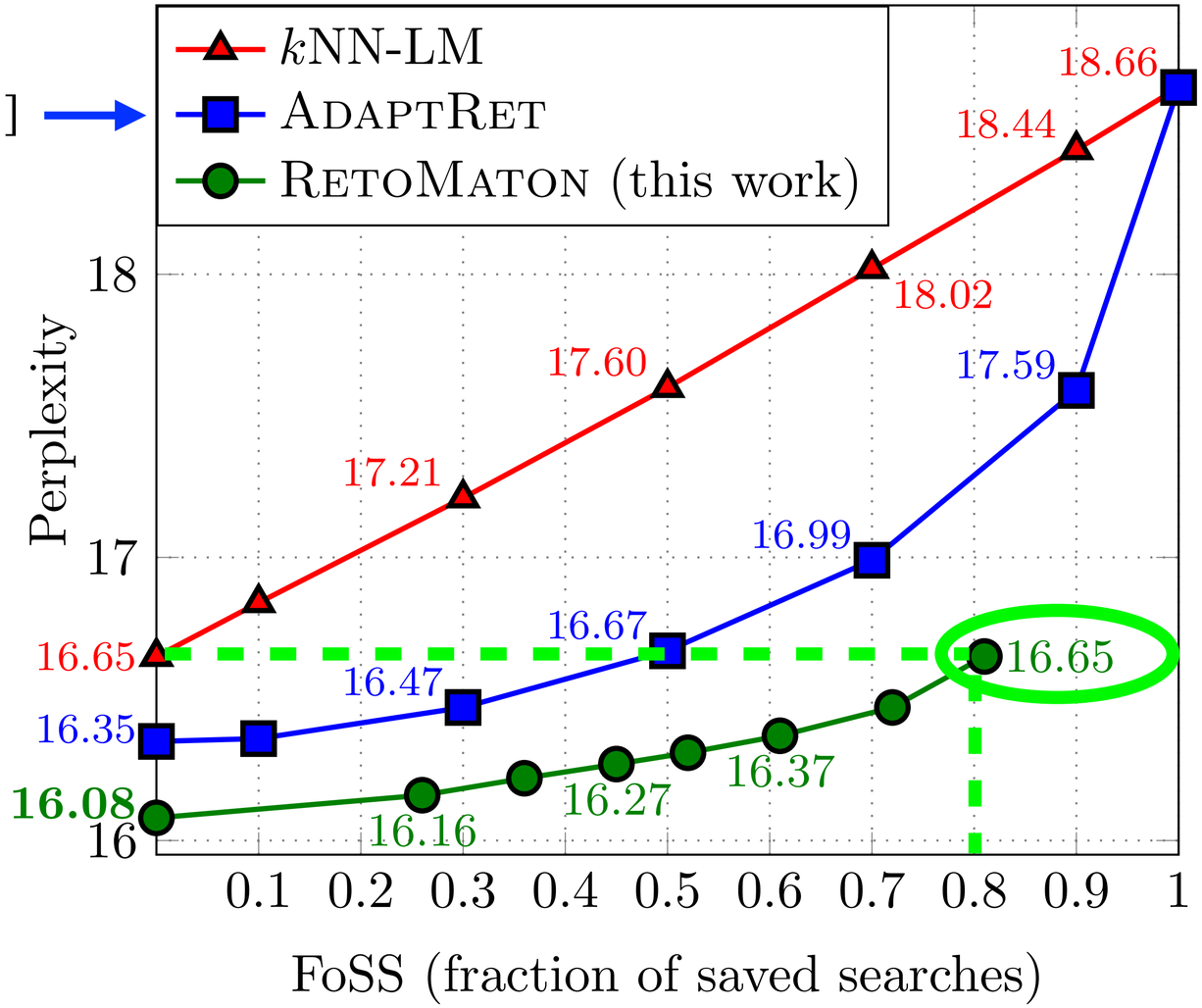
Wikitext-103

[He et al., EMNLP'2021]



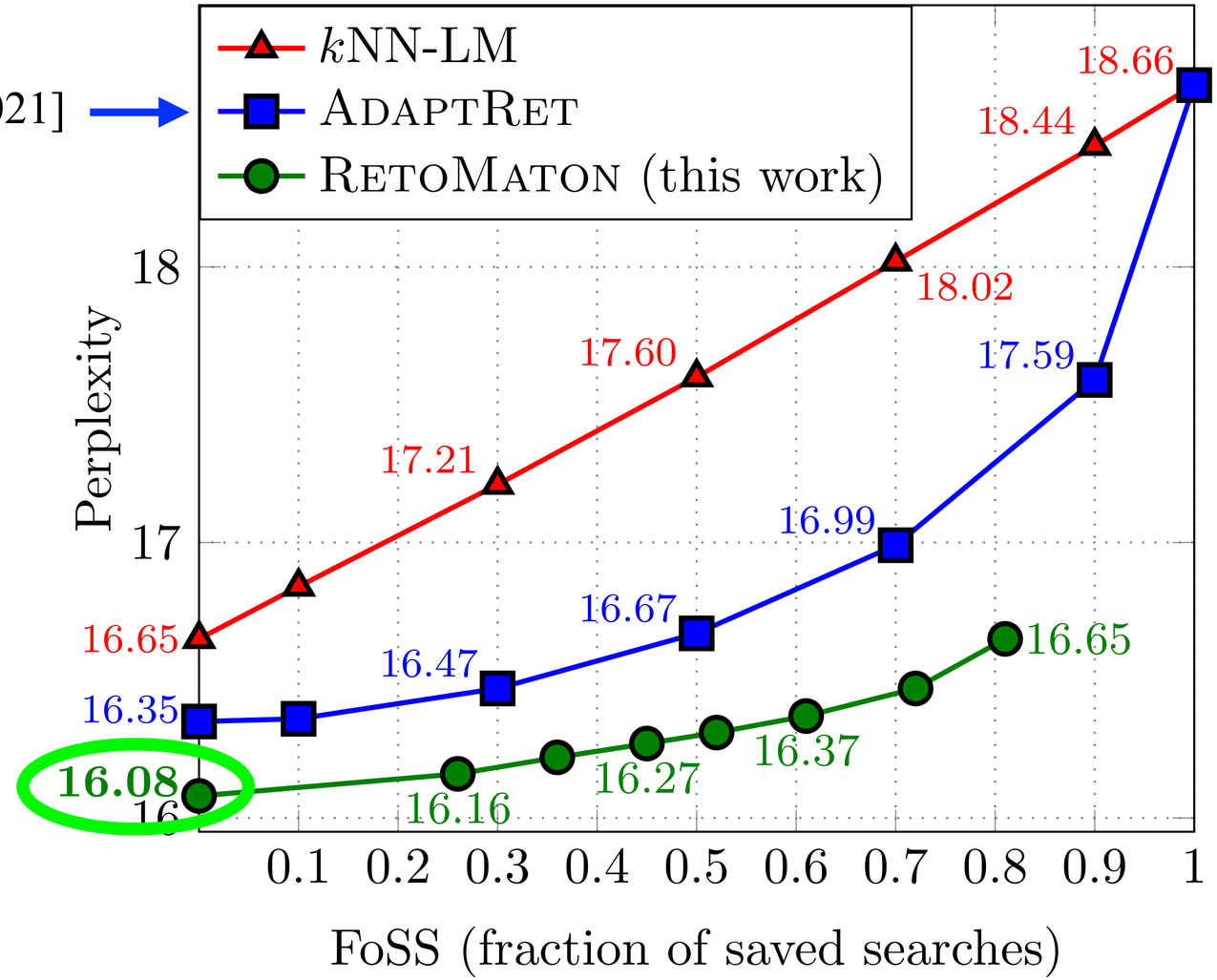
Wikitext-103

[He et al., EMNLP'2021]



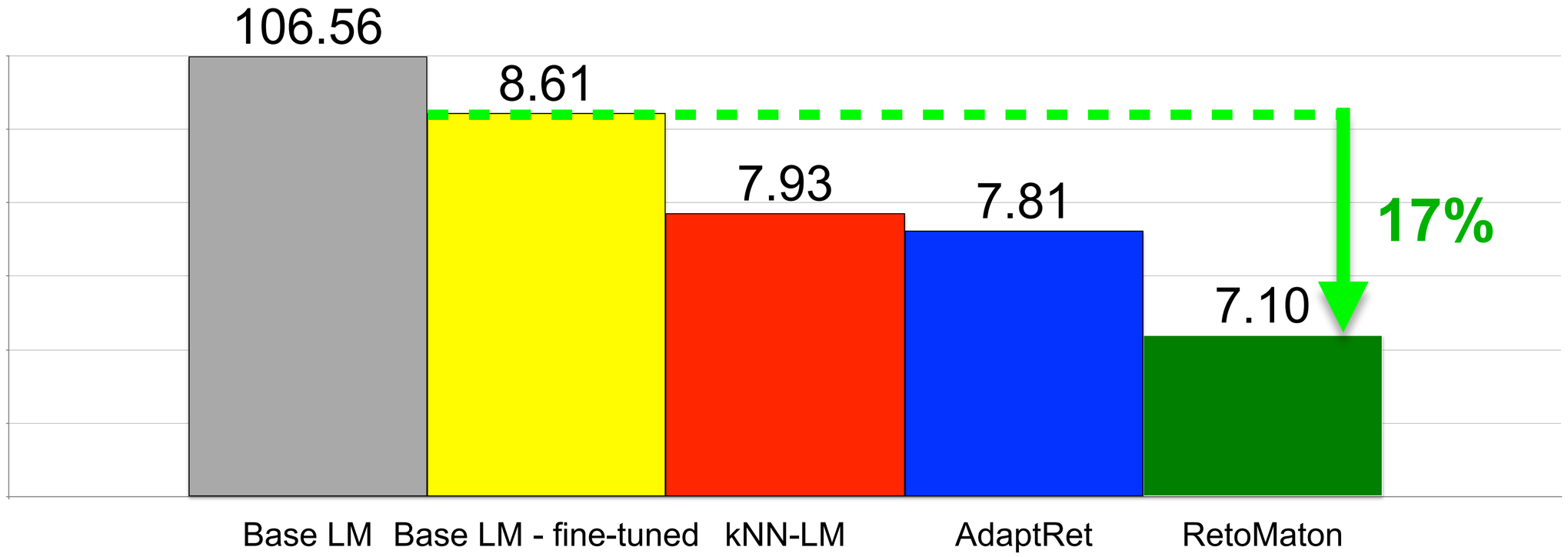
Wikitext-103

[He et al., EMNLP'2021]



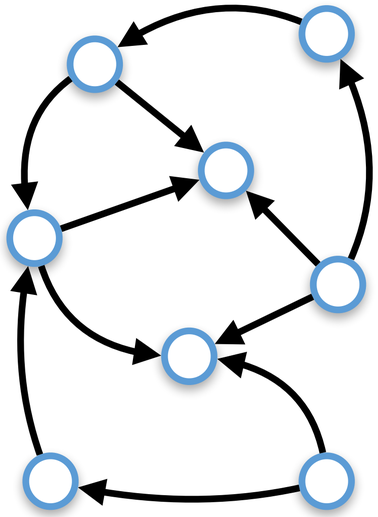
Domain Adaptation

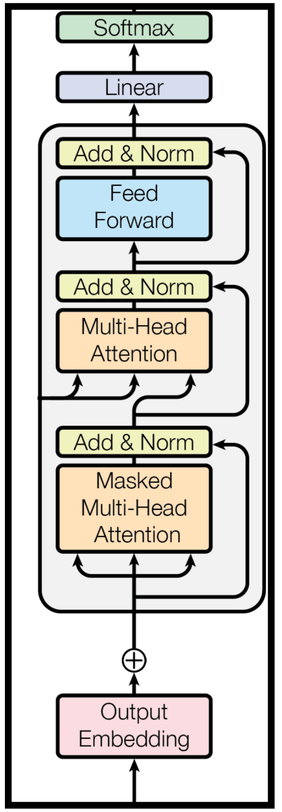
Train on WMT News Crawl; **Test**+build datastore on Law



RetoMaton

- Synergy between a **symbolic** automaton and a **neural** LM
  - Saving **pointers** between training entries
  - **Clustering** of entries into automaton states
  - ***Dynamic*** transition scores
- Lower perplexity than the base LM, while saving up to **83%** of the kNN searches compared to kNN-LM
- The creation of the automaton is **unsupervised**
  - Constructed from the original training data
  - Another domain





Please visit our poster session!

http://urialon.ml
ualon@cs.cmu.edu

https://github.com/neulab/retomaton
https://github.com/neulab/knn-transformers