Compressed Nonparametric Language Modelling

Ehsan Shareghi
Monash University

Gholamreza Haffari
Monash University

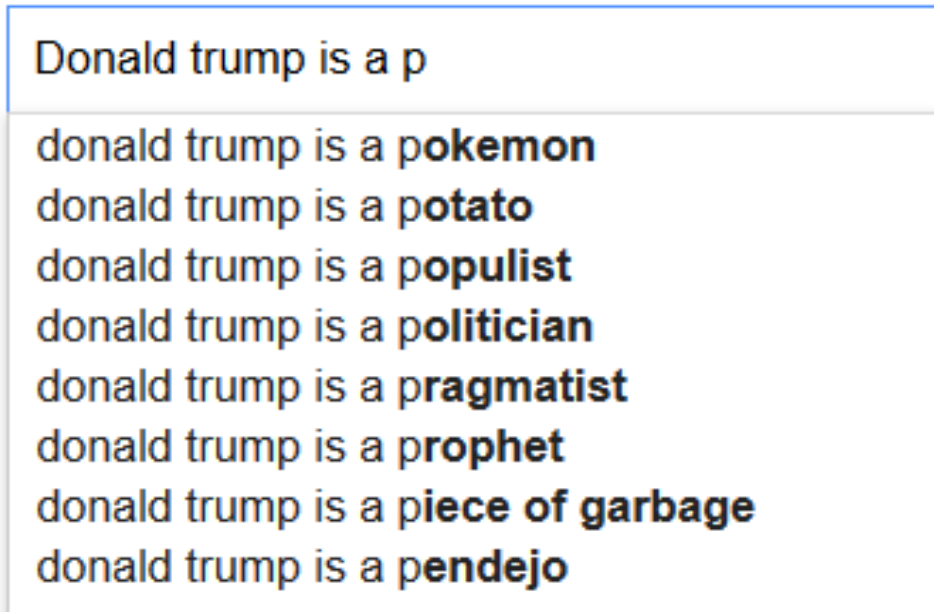
Trevor Cohn
The University of Melbourne

IJCAI 2017

Outline

- Infinite-Order Language Modelling and Challenges
- Compressed HPYP LM
- Inference and Sampling in Compressed HPYP LM
- Perplexity and Mixing
- Conclusion and Future Directions

Language Modelling (LM)



Predictive typing/Auto completion

Président de la Chambre des représentants

President of the Bedroom of Representatives
President of the House of Representatives

P(House | President of the) > P(Bedroom | President of the)

Machine Translation

Infinite order LM

$$P(w_1^N) = \prod_{i=1}^{N} P(w_i | w_1^{i-1})$$

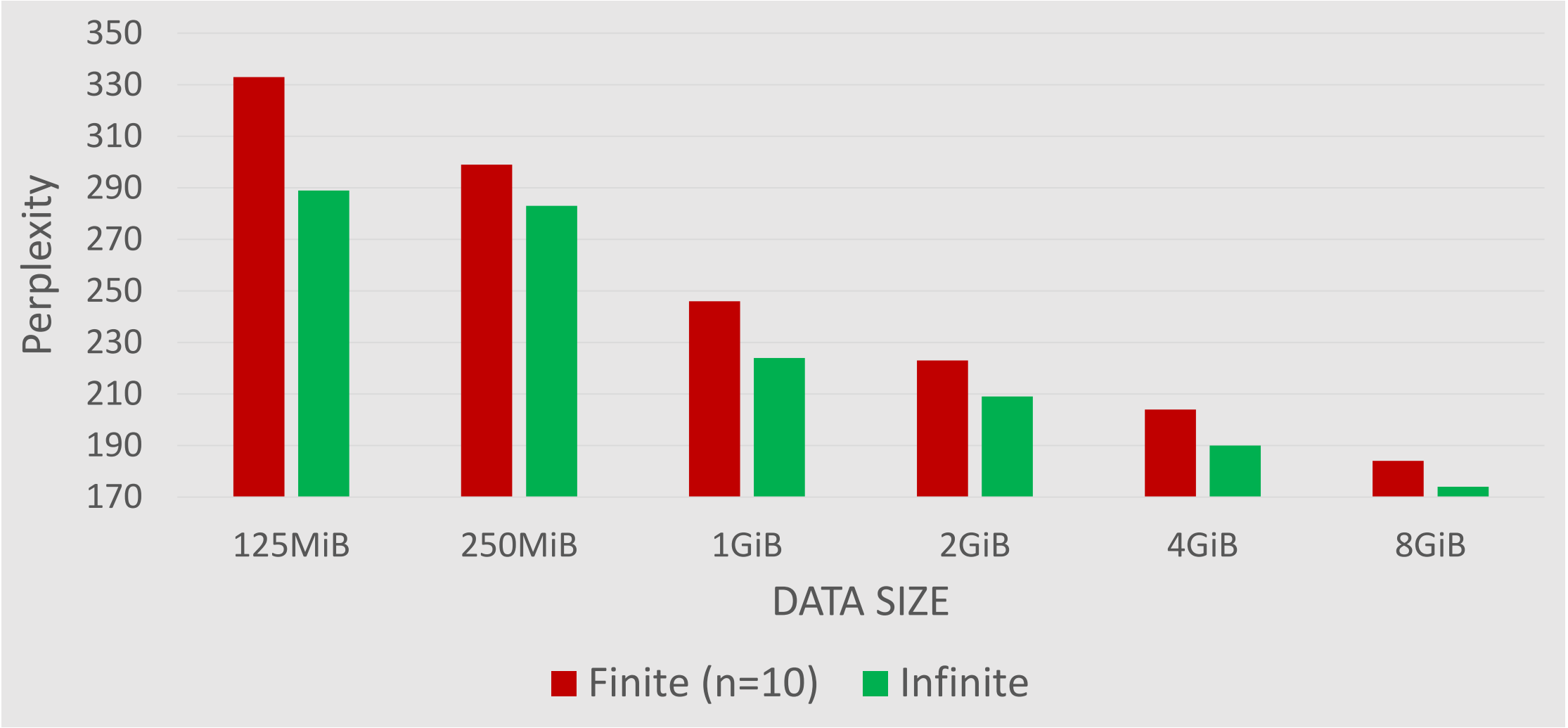
Statistical sparsity

Solution: smoothing (HPYP, etc)

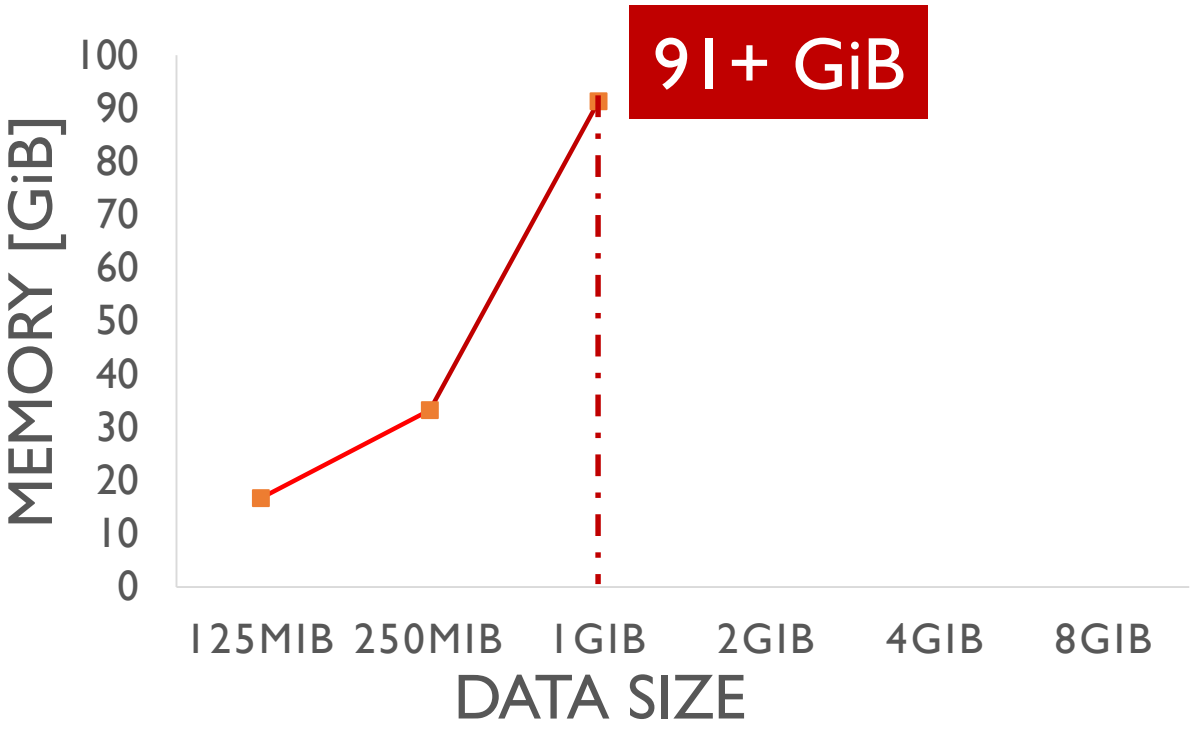
Computational cost of smoothing

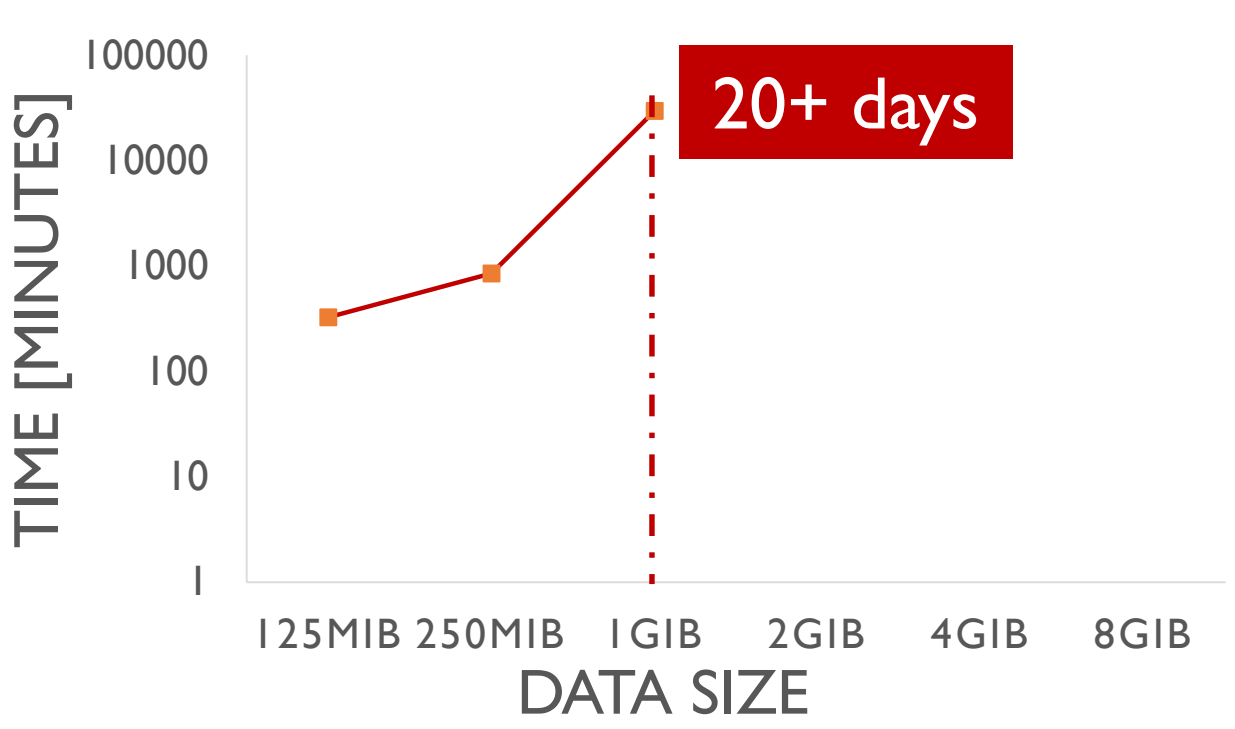
No Scalable Solution

Why Infinite-order LM ?



Computational Cost of HPYP LM - Training





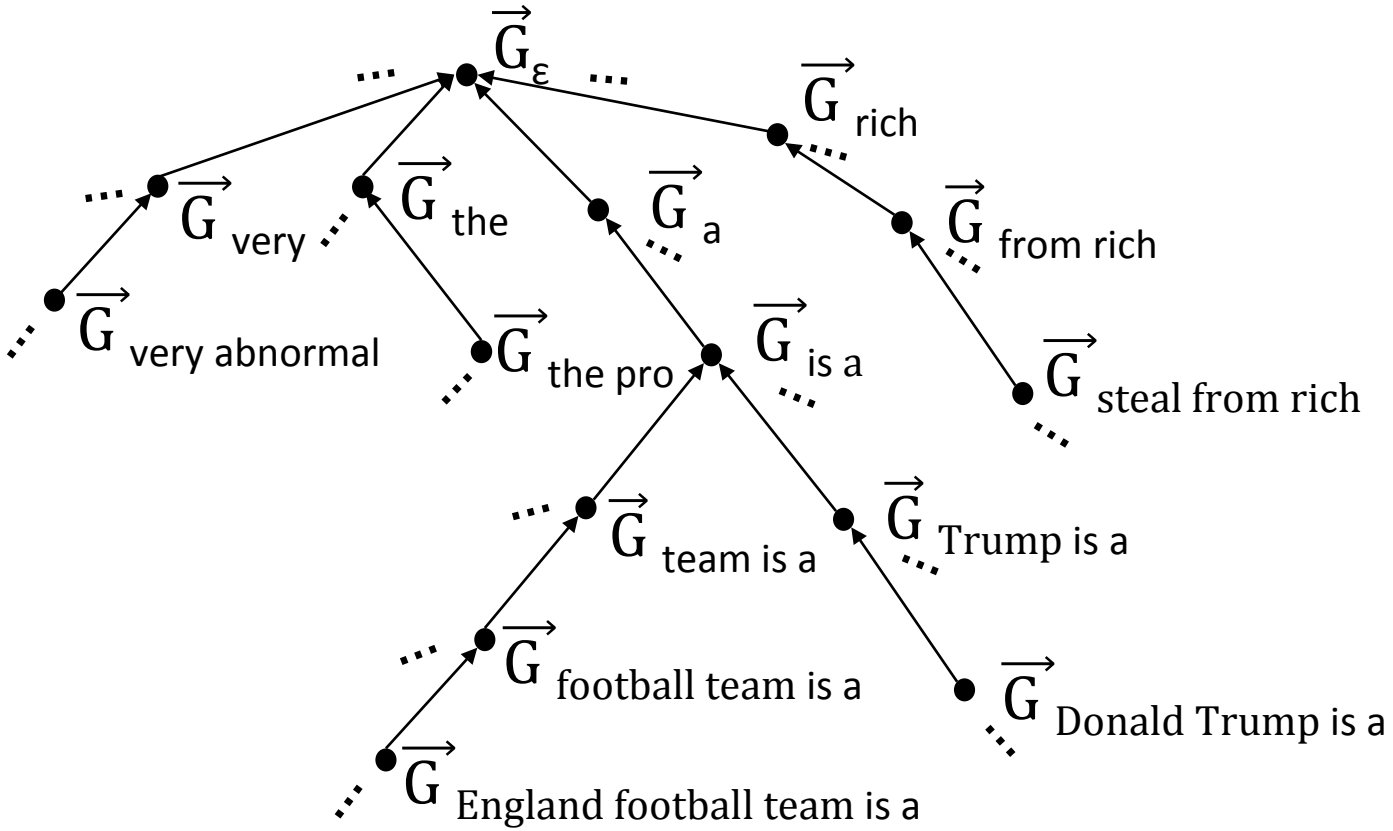
Involved Factors:

- Building Model (hierarchy)
- Parameters Sampling
- Storing the Model and Parameters

Outline

- Infinite-Order Language Modelling and Challenges
- Compressed HPYP LM
- Inference and Sampling in Compressed HPYP LM
- Perplexity and Mixing
- Conclusion and Future Directions

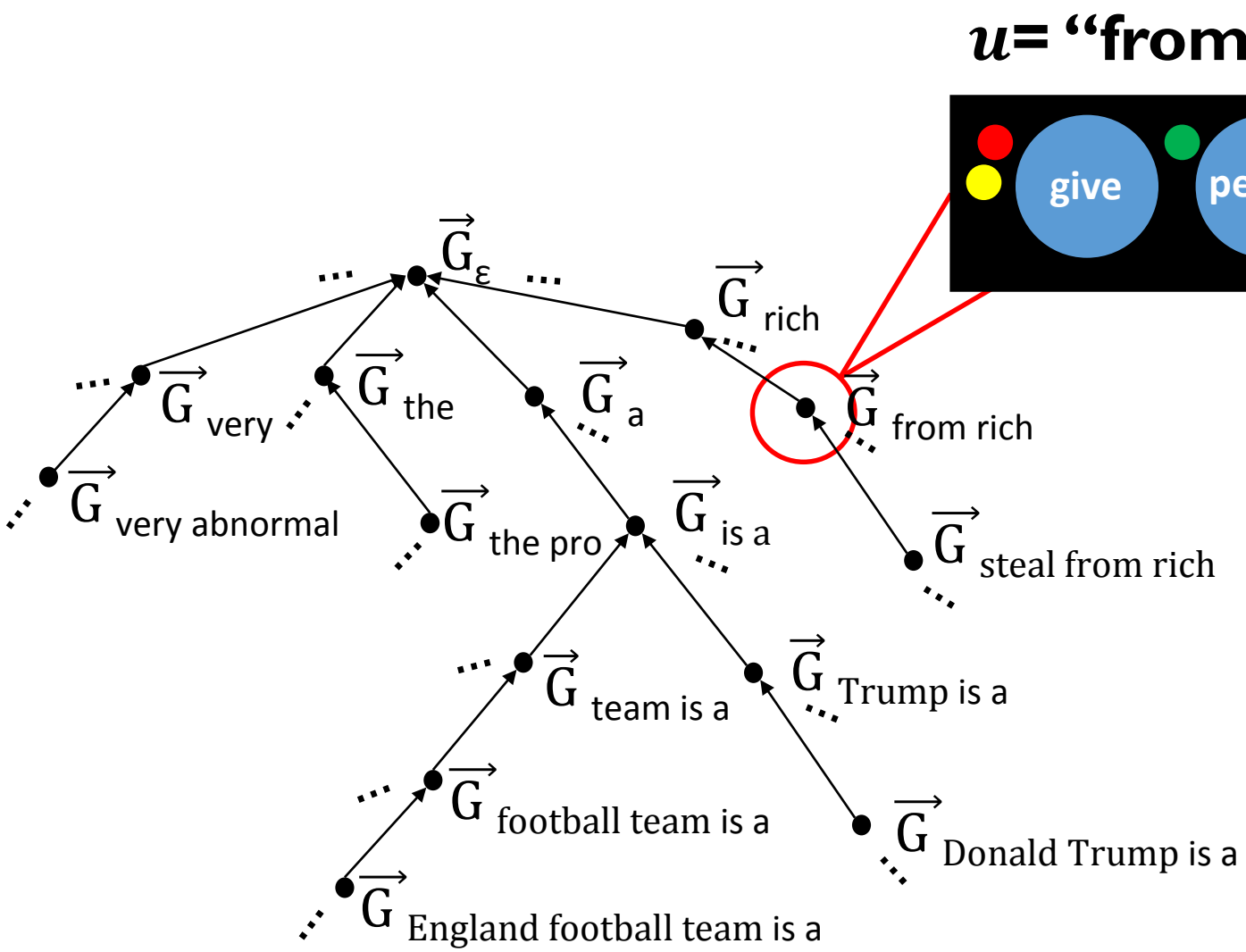
Hierarchical Pitman-Yor Process (HPYP)



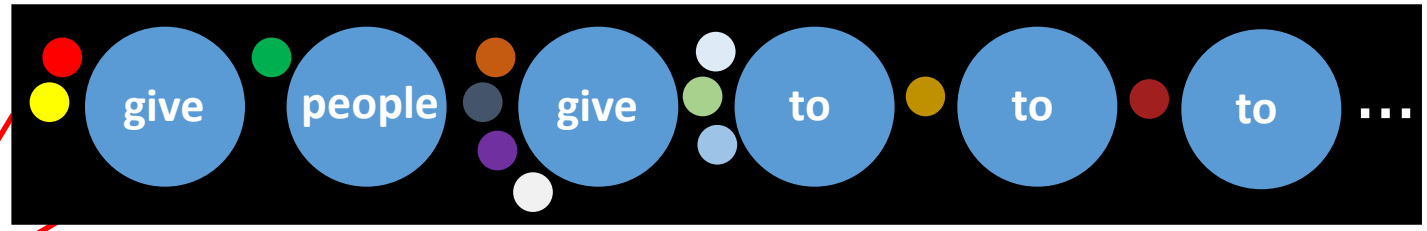
$$\vec{G}_u \sim \mathrm{PYP}(\theta_u, d_u, \vec{G}_{\pi(u)})$$
$$\vec{G}_\varepsilon \sim \mathrm{PYP}(\theta_\varepsilon, d_\varepsilon, \frac{1}{|\mathrm{vocab}|})$$

Same model as "Sequence Memoizer" Wood et al. (2011)

HPYP LM – Chinese Restaurant Process

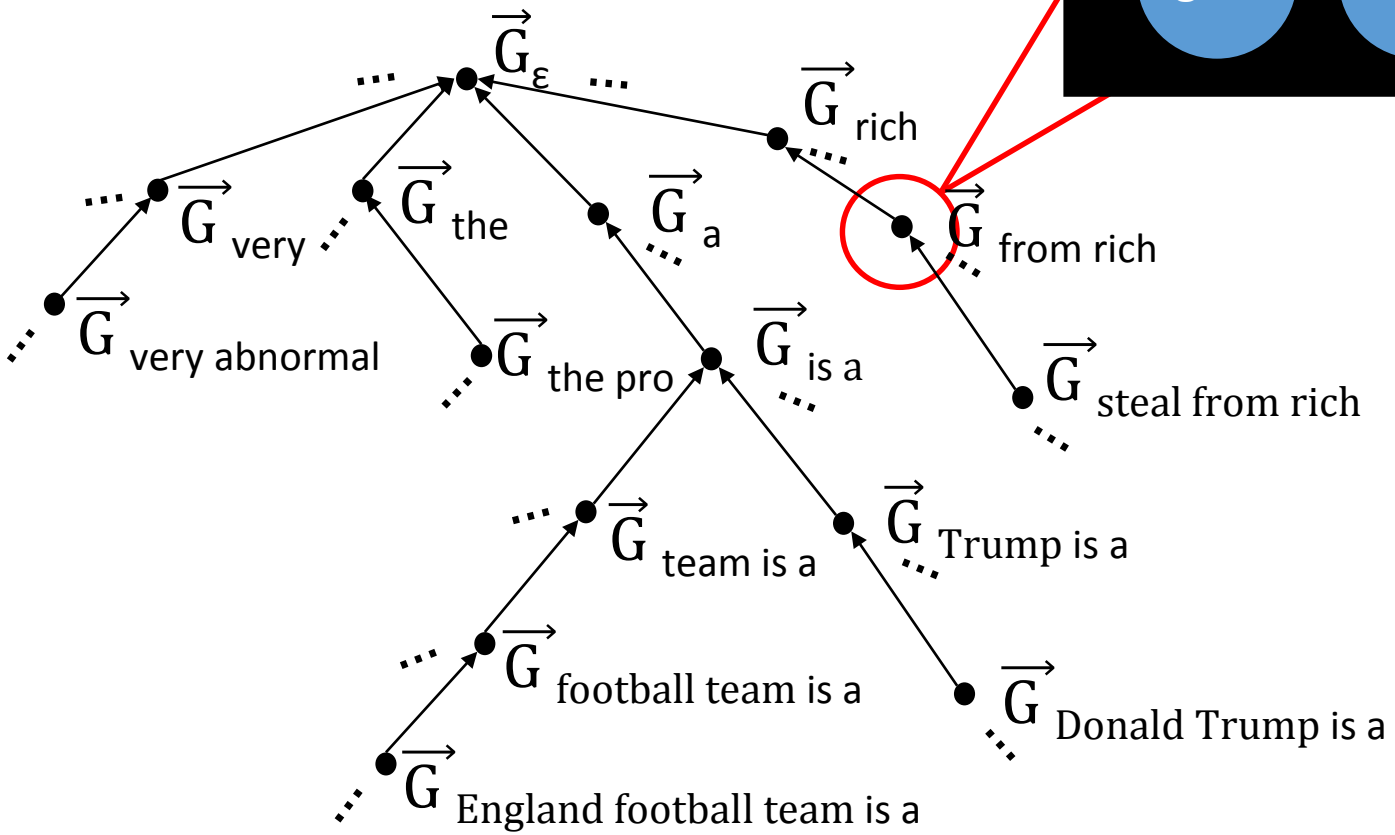


u= "from rich", $\eta_u = \{d_u, \theta_u, \{n_u^w, t_u^w\}_{w \in u}\}$

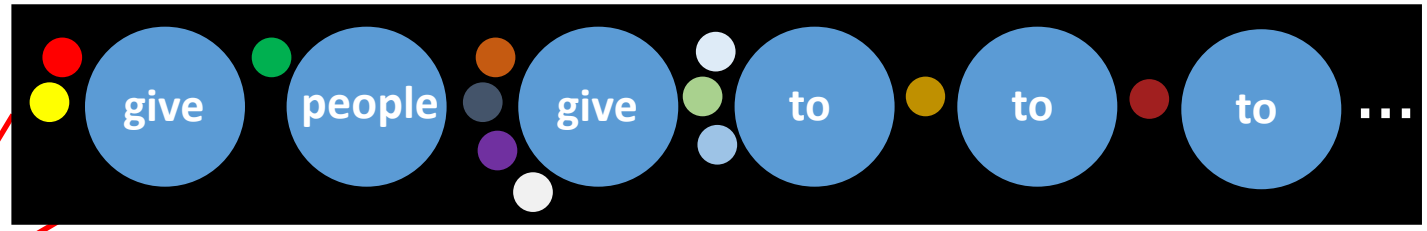


$n_u^{give} = 6$, $t_u^{give} = 2$

HPYP LM – Chinese Restaurant Process



u= "from rich", $\eta_u = \{d_u, \theta_u, \{n_u^w, t_u^w\}_{w \in u}\}$



$n_u^{give} = 6\ ,\ t_u^{give} = 2$

$$0 \le t_u^w \le n_u^w$$

$$n_{\pi(u)}^w = \sum_{v \in \text{children}(\pi(u))} t_v^w$$

Compressed HPYP LM

- Hierarchy of KN and HPYP LMs are the same
- KN can serve as an approximate inference for HPYP ($\theta_u = 0$ and $t_u^w = 1$)

Compressed HPYP LM

- Hierarchy of KN and HPYP LMs are the same
- KN can serve as an approximate inference for HPYP ($\theta_u = 0$ and $t_u^w = 1$)

- KN hierarchy can be recovered from a compressed suffix tree of data on-the-fly
- Compressed Suffix Trees:
  - Based on advanced data structures such as the Wavelet Tree of the BWT of text
  - Contain all the information about the HPYP hierarchy and text itself in a space matching the text size

Shareghi et al., 2016, *Fast, Small and Exact Language Modelling with Compressed Suffix Trees*, *Transactions of the ACL*

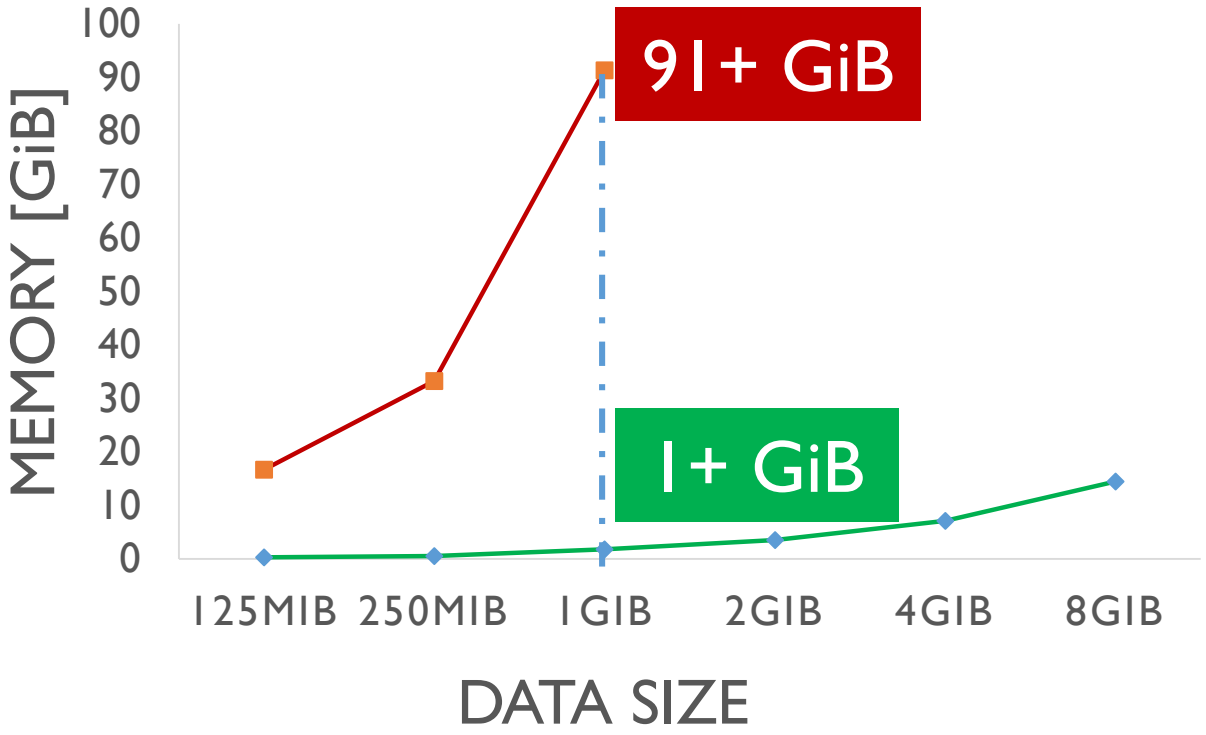
Compressed HPYP LM

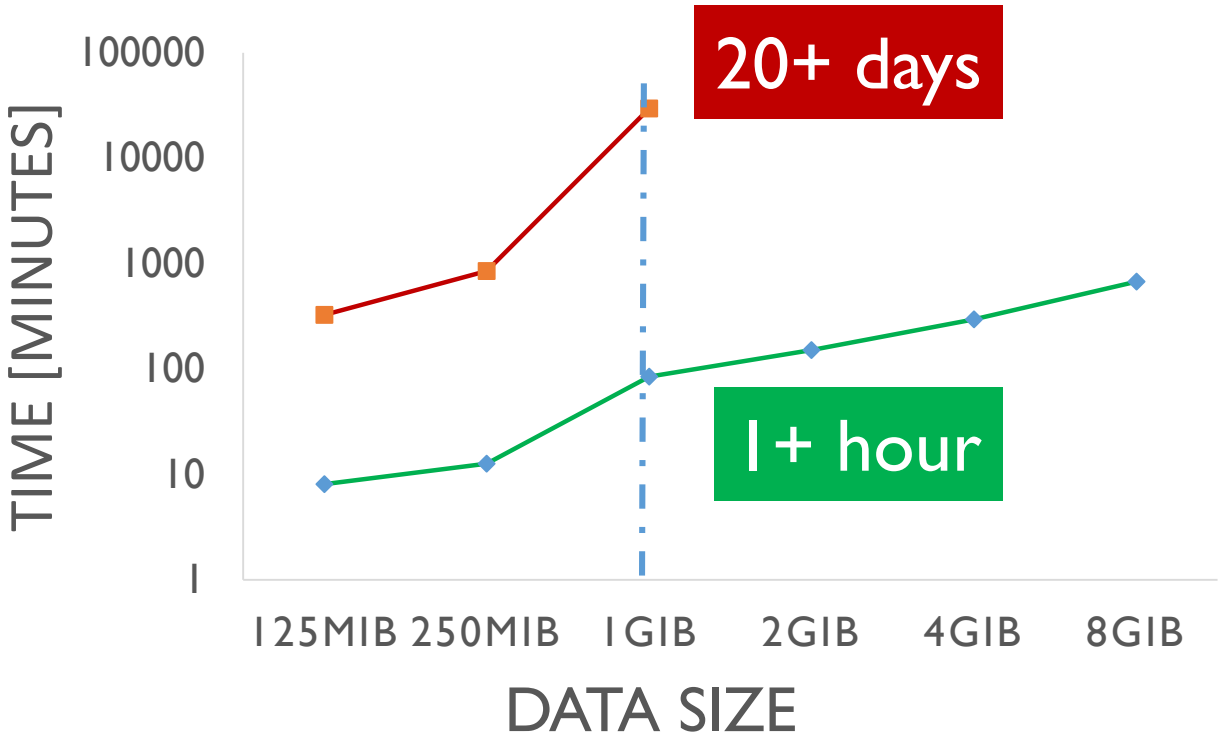
- Hierarchy of KN and HPYP LMs are the same
- KN can serve as an approximate inference for HPYP ($\theta_u = 0$ and $t_u^w = 1$)

- KN hierarchy can be recovered from a compressed suffix tree of data on-the-fly
- Compressed Suffix Trees:
  - Based on advanced data structures such as the Wavelet Tree of the BWT of text
  - Contain all the information about the HPYP hierarchy and text itself in a space matching the text size

Compressed HPYP	HPYP
Constructs Compressed Suffix Tree of Data	Constructs Hierarchy of HPYP
No Sampling	Samples across all nodes and for all w

Training time comparison





Outline

- Infinite-Order Language Modelling and Challenges
- Compressed HPYP LM
- Inference and Sampling in Compressed HPYP LM
- Perplexity and Mixing
- Conclusion and Future Directions

Inference in Compressed HPYP

We need to compute the following **intractable** integral,

$$P(w|u) = \int P(w|u, \eta)\, P(\eta)\, d\eta$$

which we approximate using samples for $\eta_u = \{d_u, \theta_u, \{\boldsymbol{n_u^w}, \boldsymbol{t_u^w}\}_{w \in u}\}$.

$$0 \leq t_u^w \leq n_u^w$$

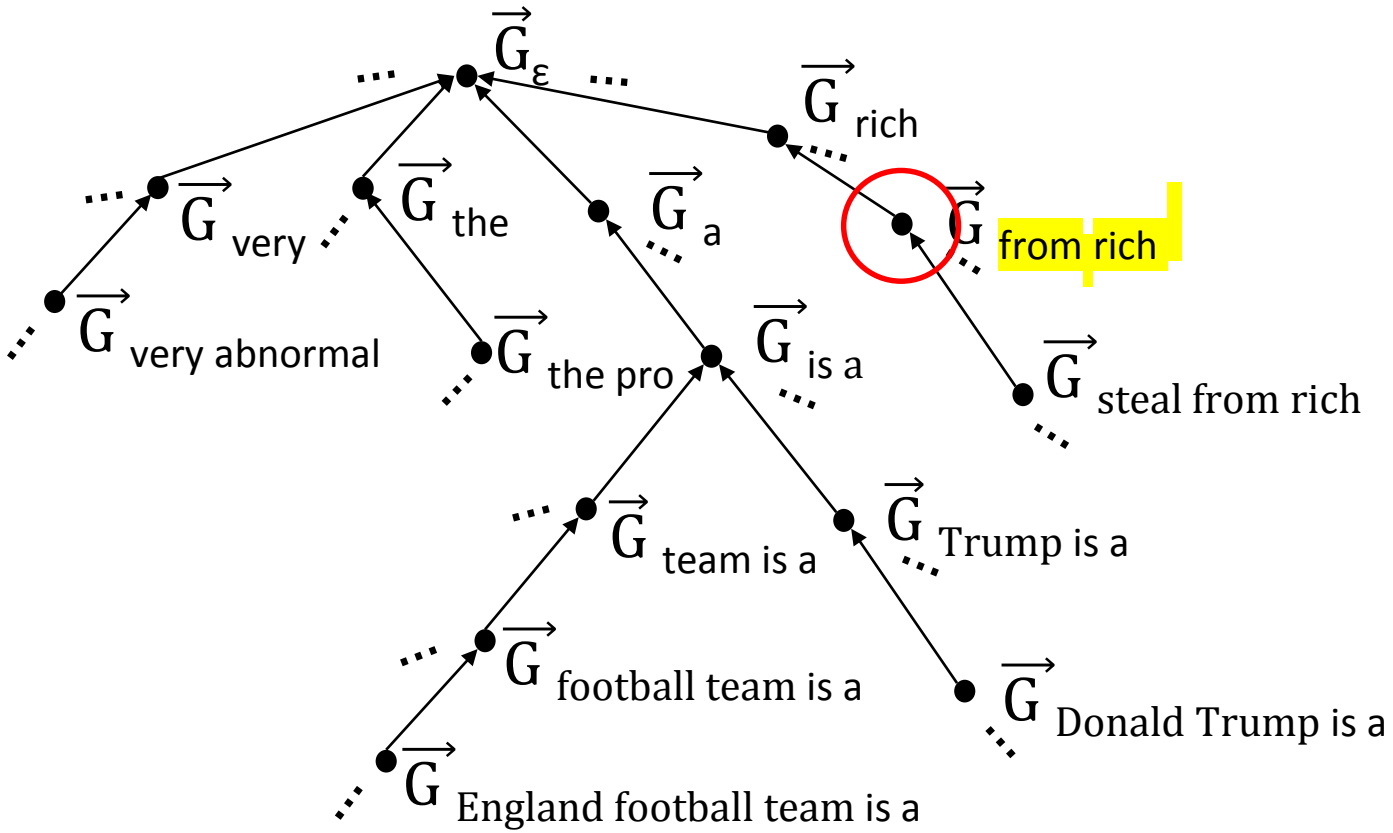
$$n_{\pi(u)}^w = \sum_{v \in \text{children}(\pi(u))} \boldsymbol{t_v^w}$$

Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)

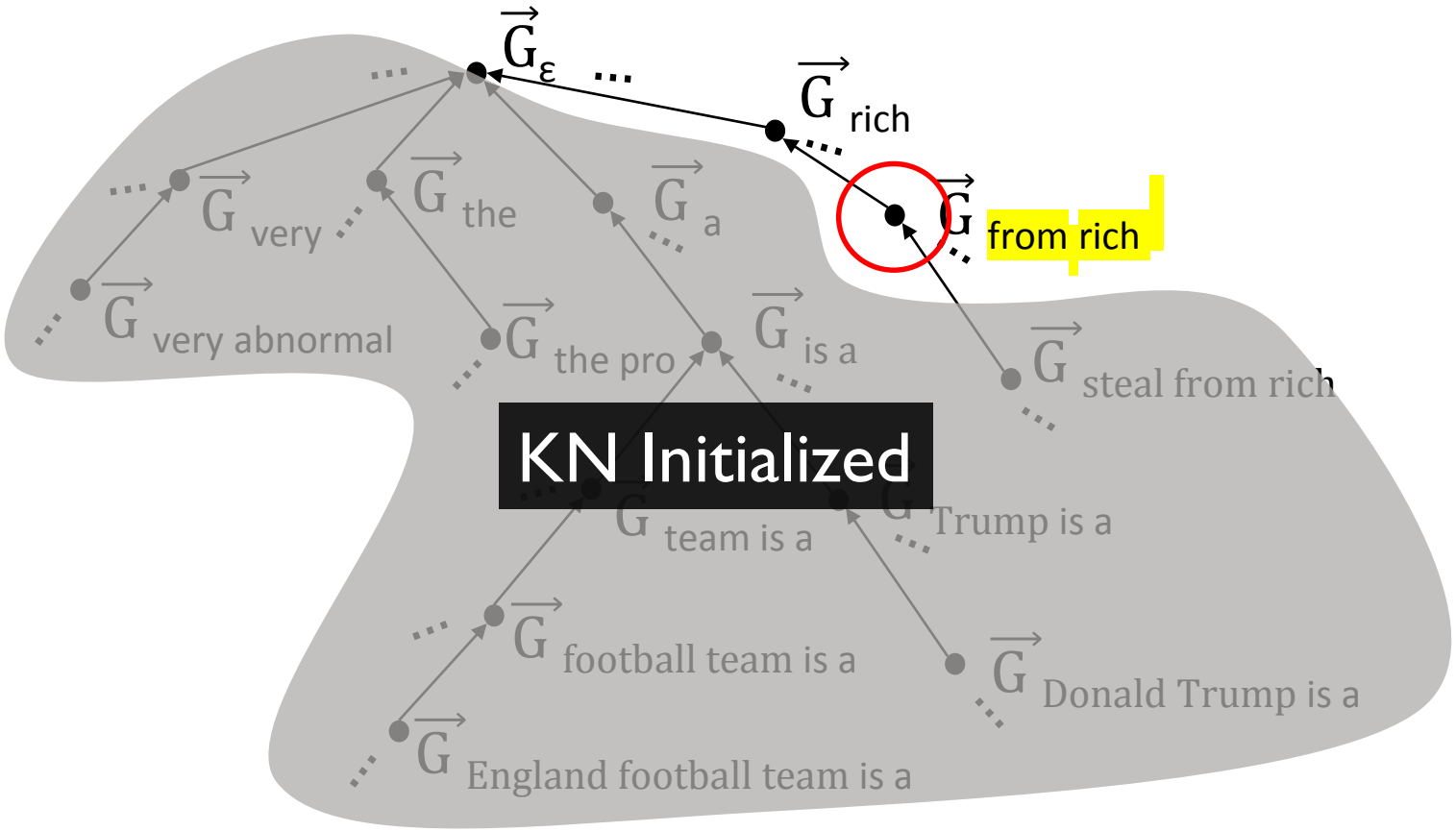
Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)



Sampling $\{n_u^w, t_u^w\}_{w \in u}$

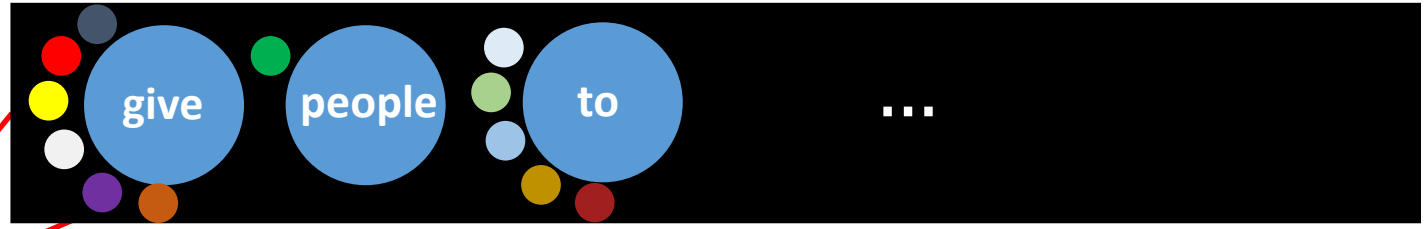
Given a query P(give | from rich)

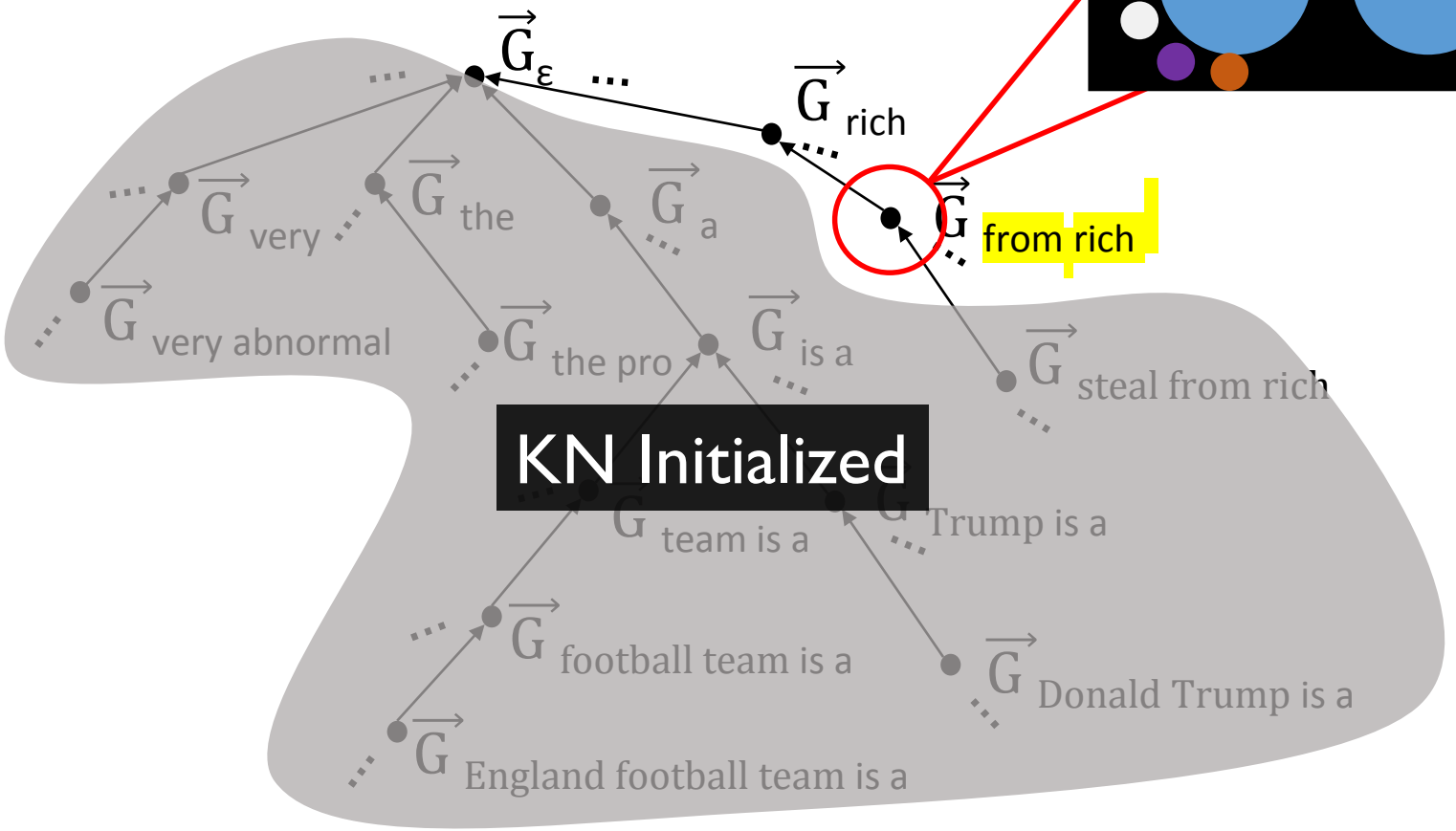


Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)

Read $n^{\text{give}}_{\text{from rich}}$ from Data

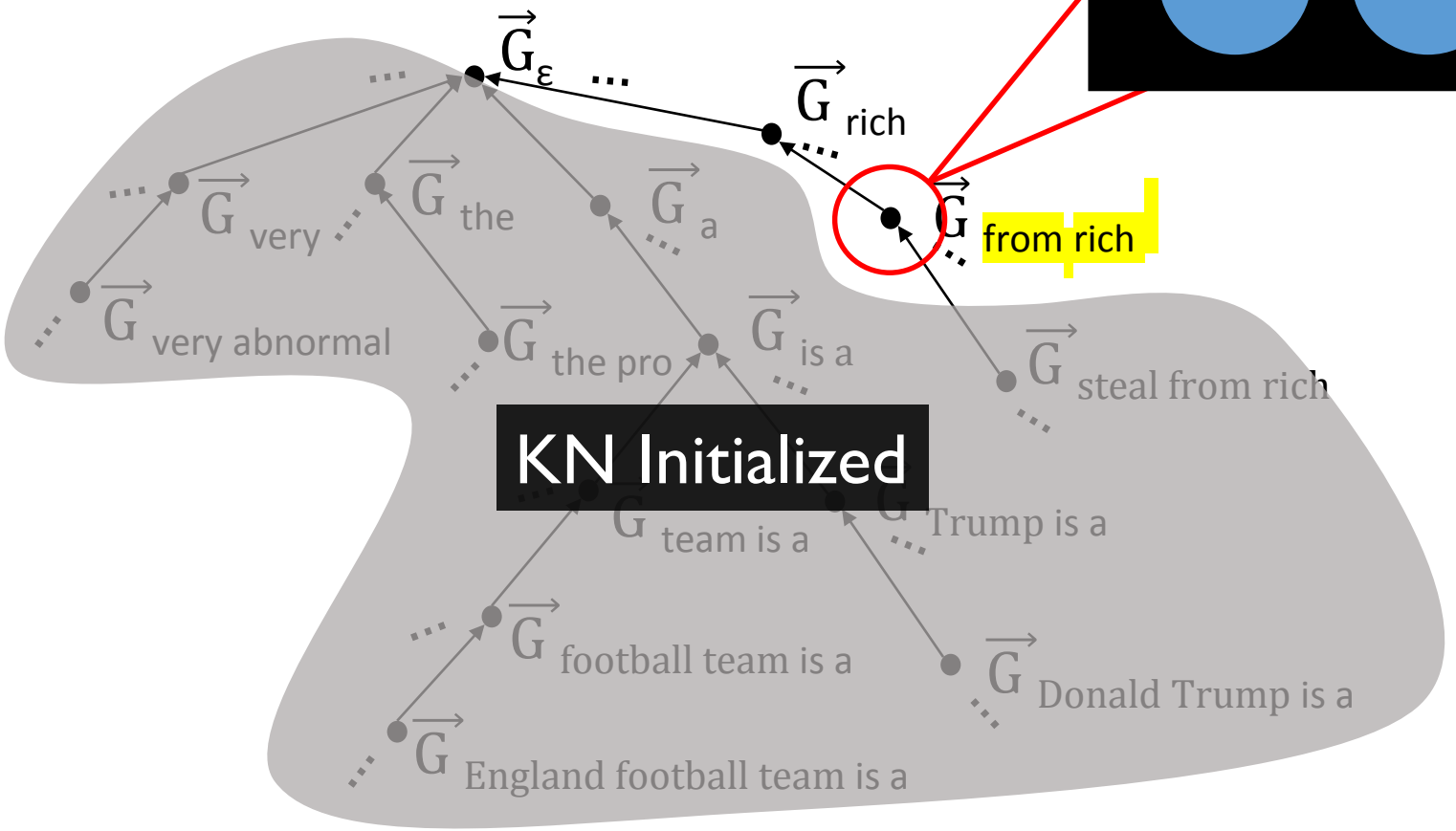




Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)

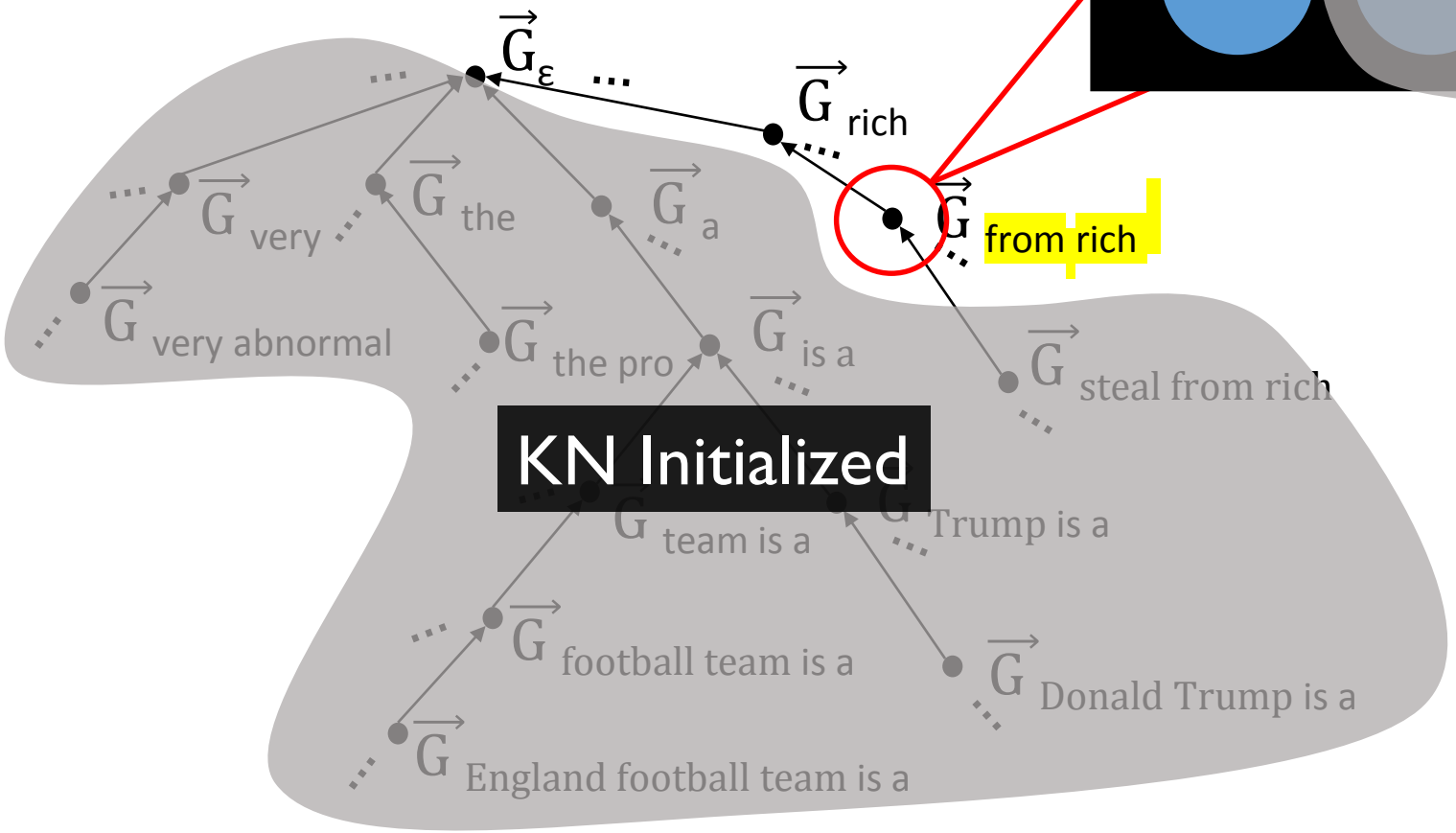
Sample $t^{\text{give}}_{\text{from rich}} = 2$



Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)

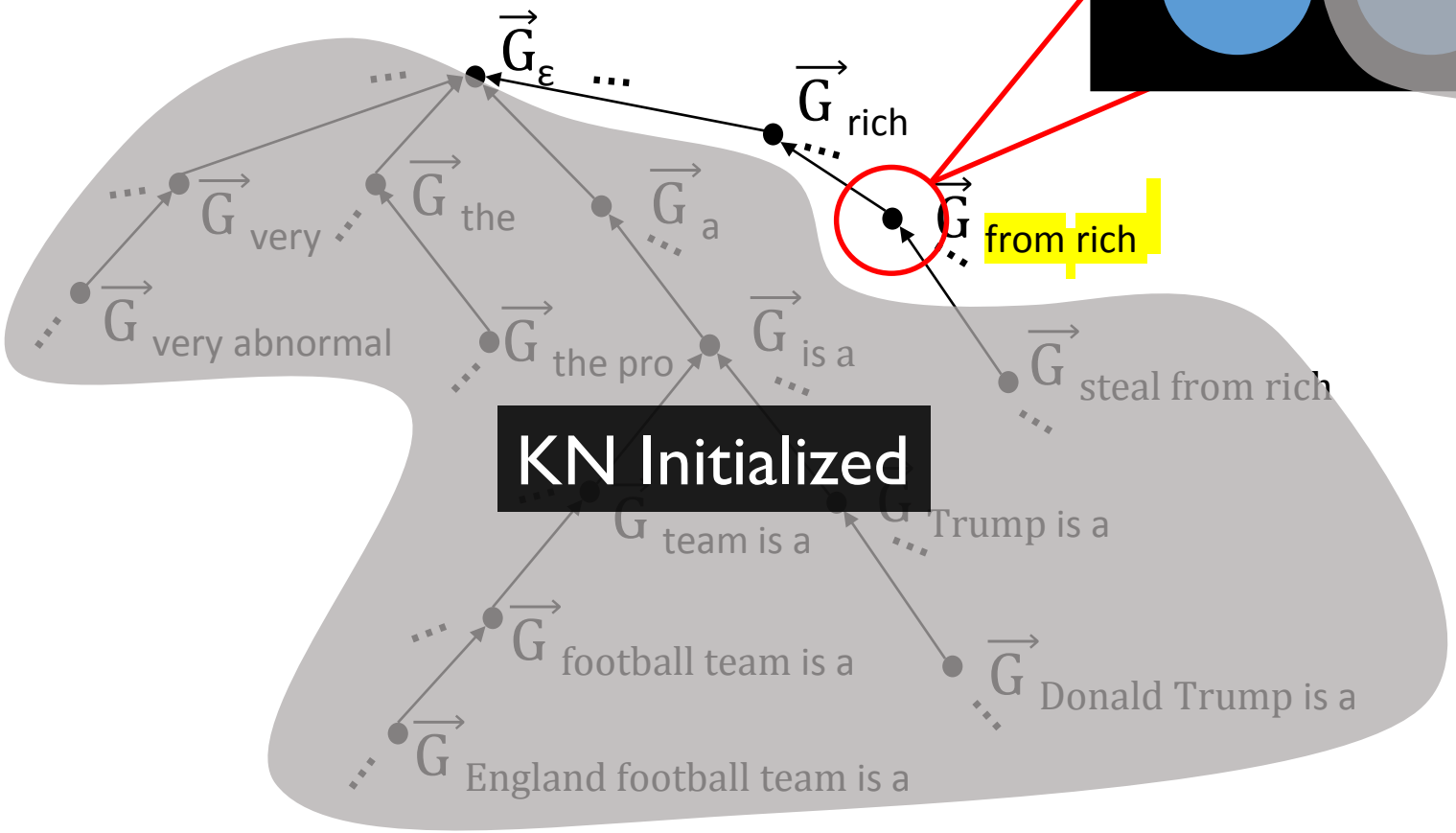
Sample $t^{\text{give}}_{\text{from rich}} = 2$



Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)

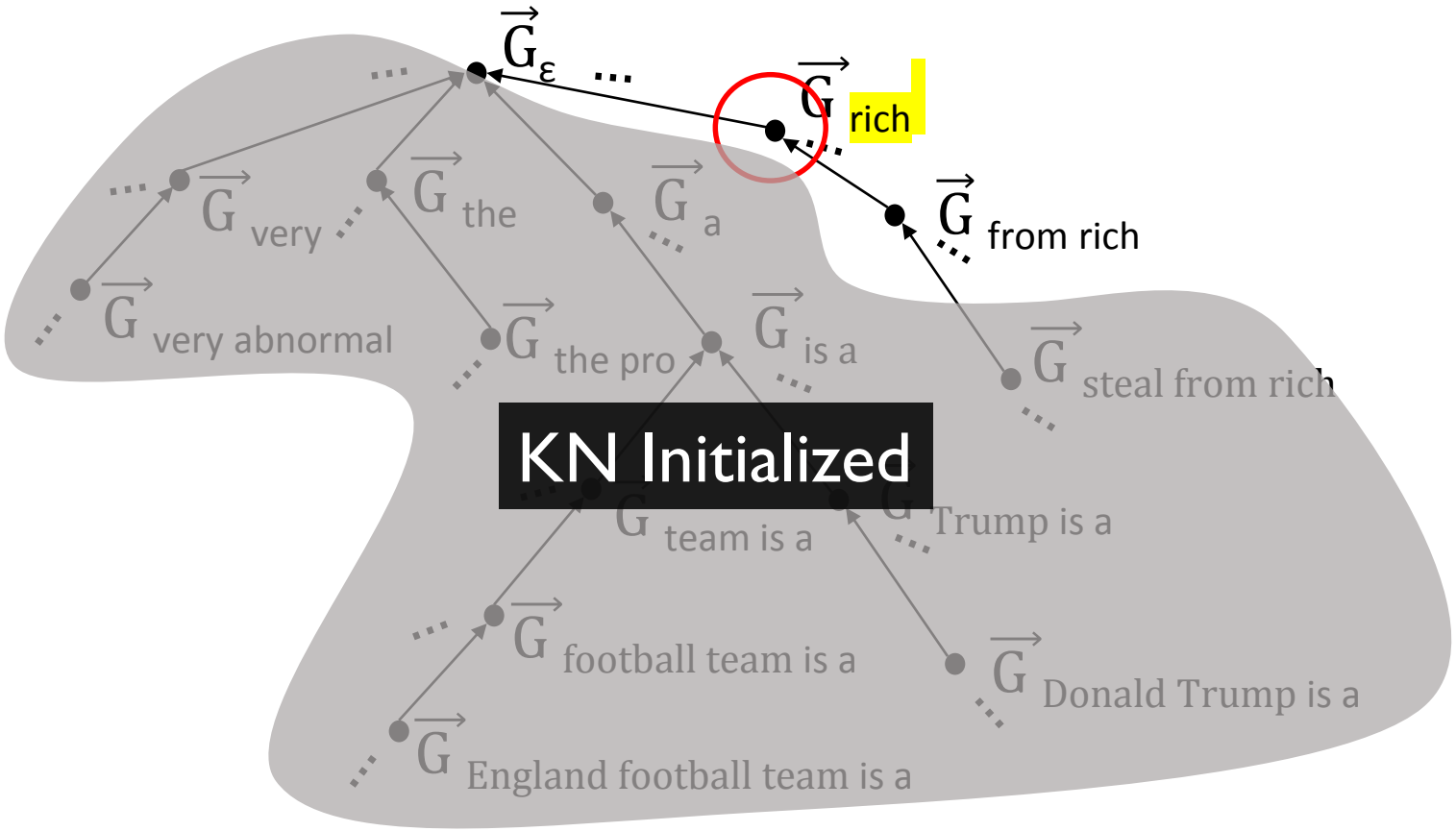
Update $n_{\text{rich}}^{\text{give}}$



KN Initialized

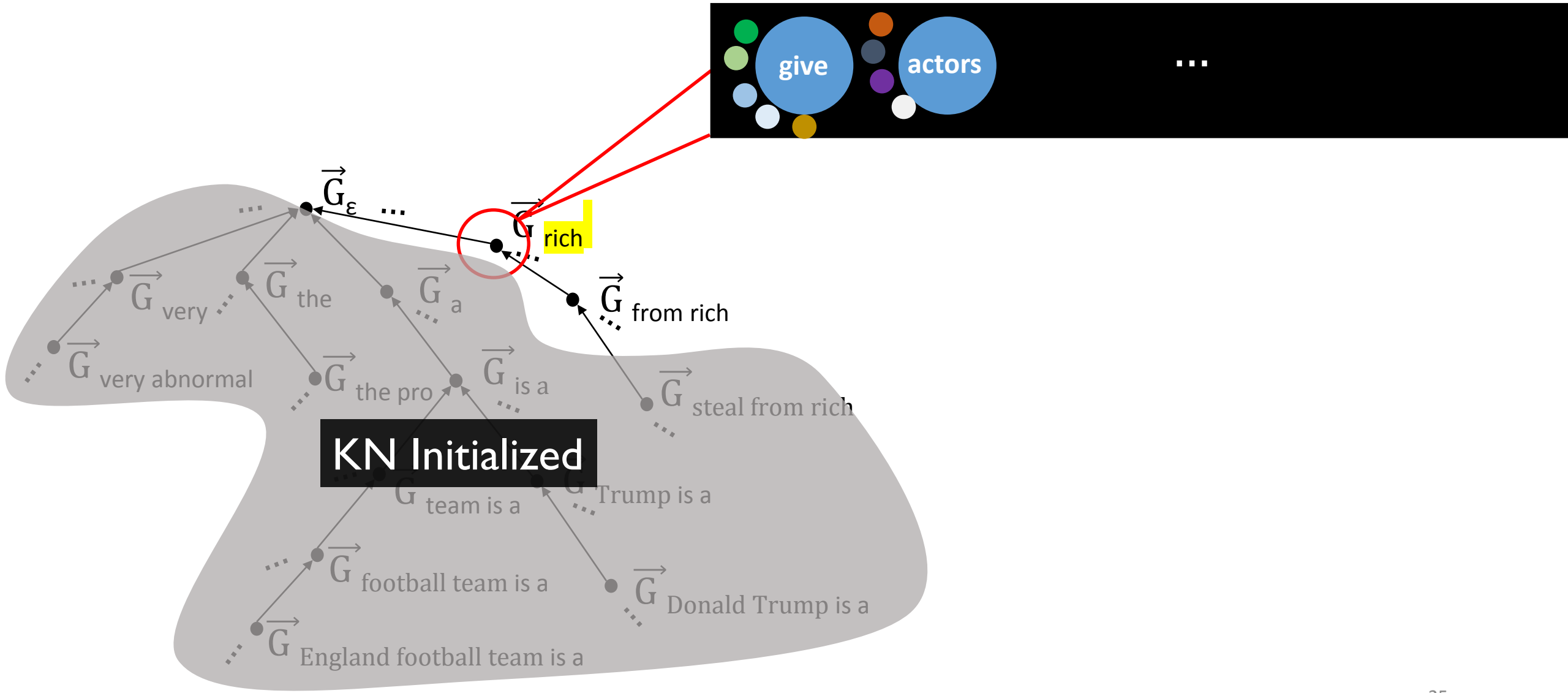
Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)



Sampling $\{n_u^w, t_u^w\}_{w \in u}$

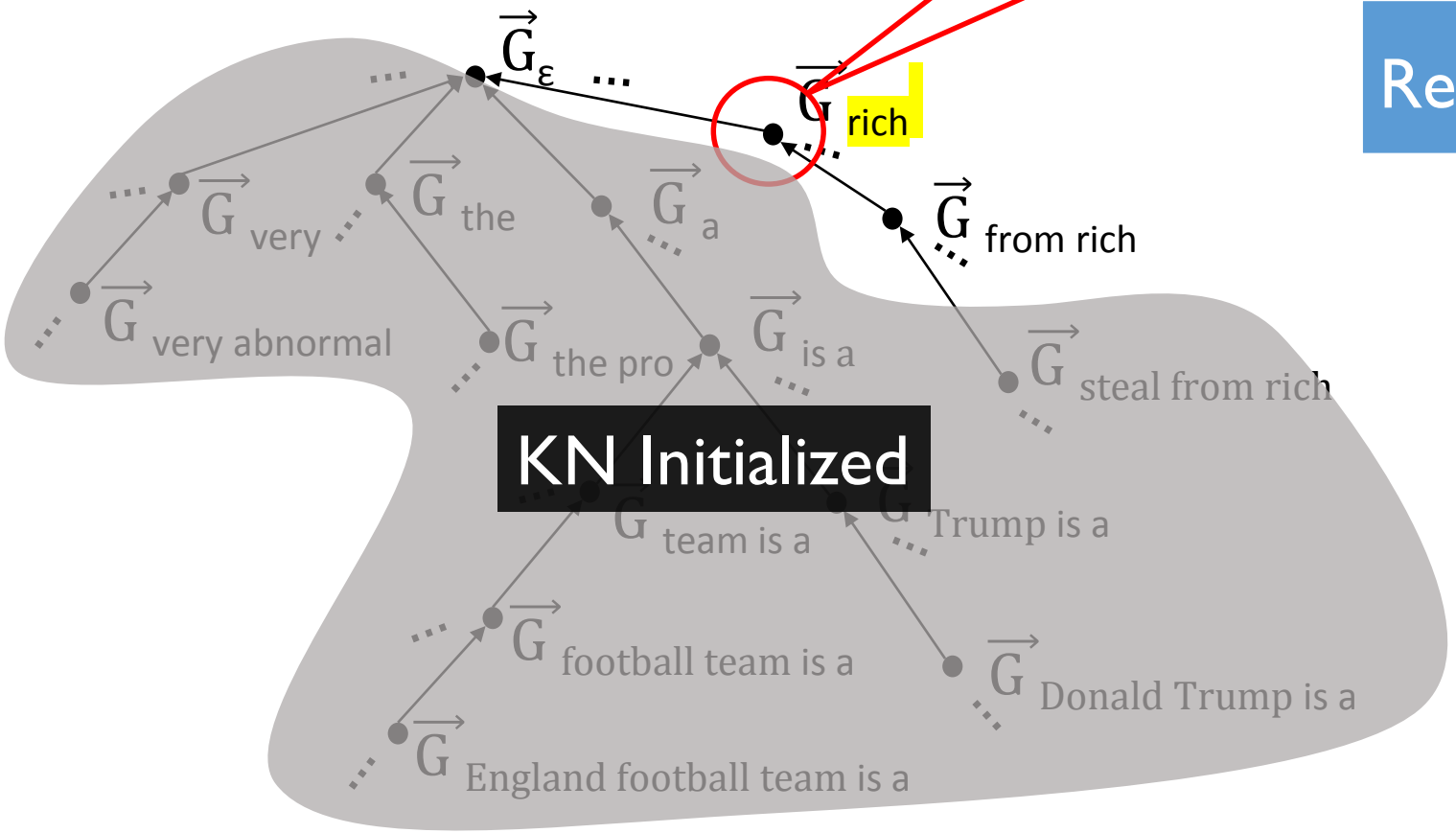
Given a query P(give | from rich)



Sampling $\{n_u^w, t_u^w\}_{w \in u}$

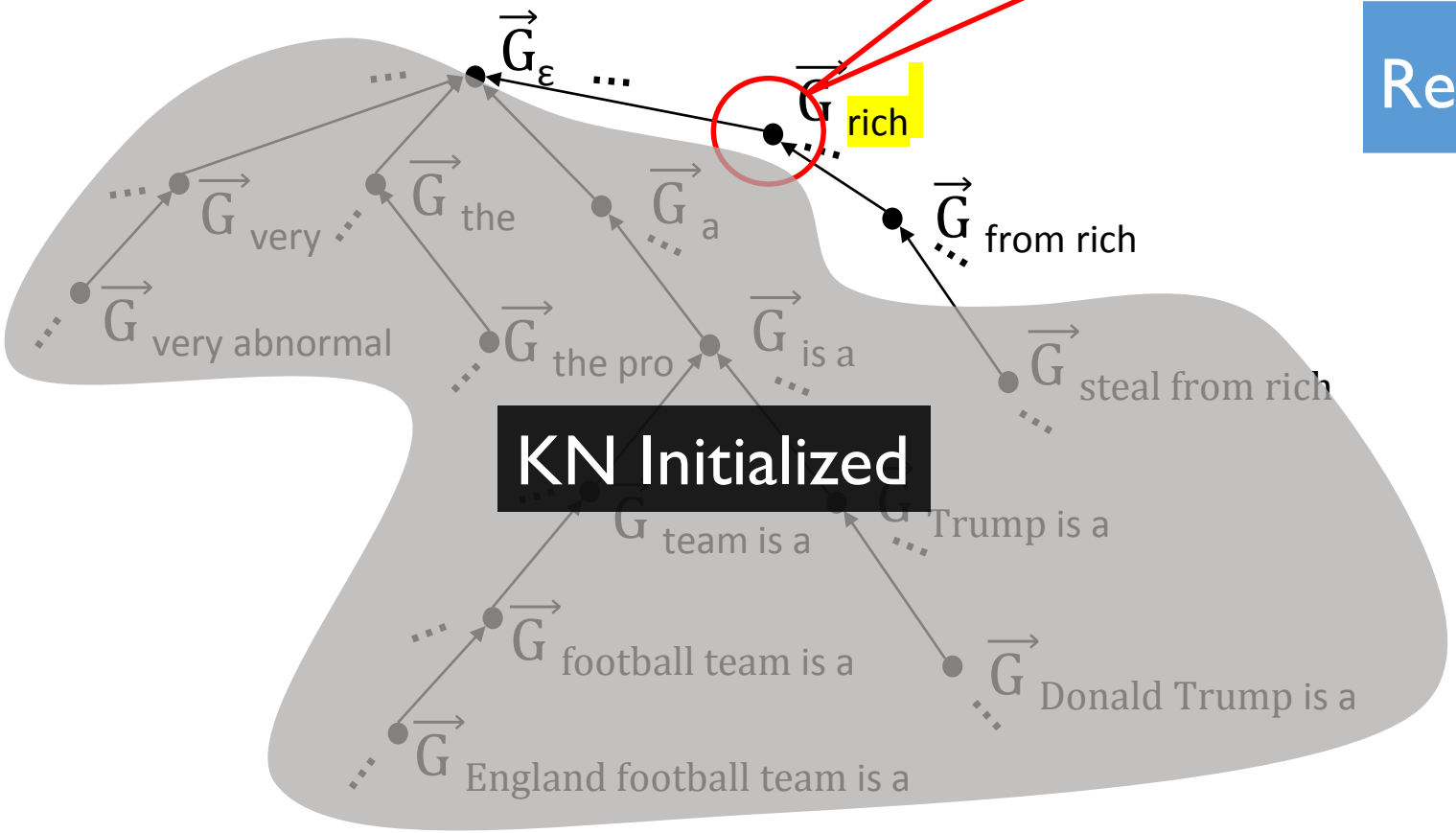
Given a query P(give | from rich)

Read $n_{\text{rich}}^{\text{give}}$ from proxy counts



Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)

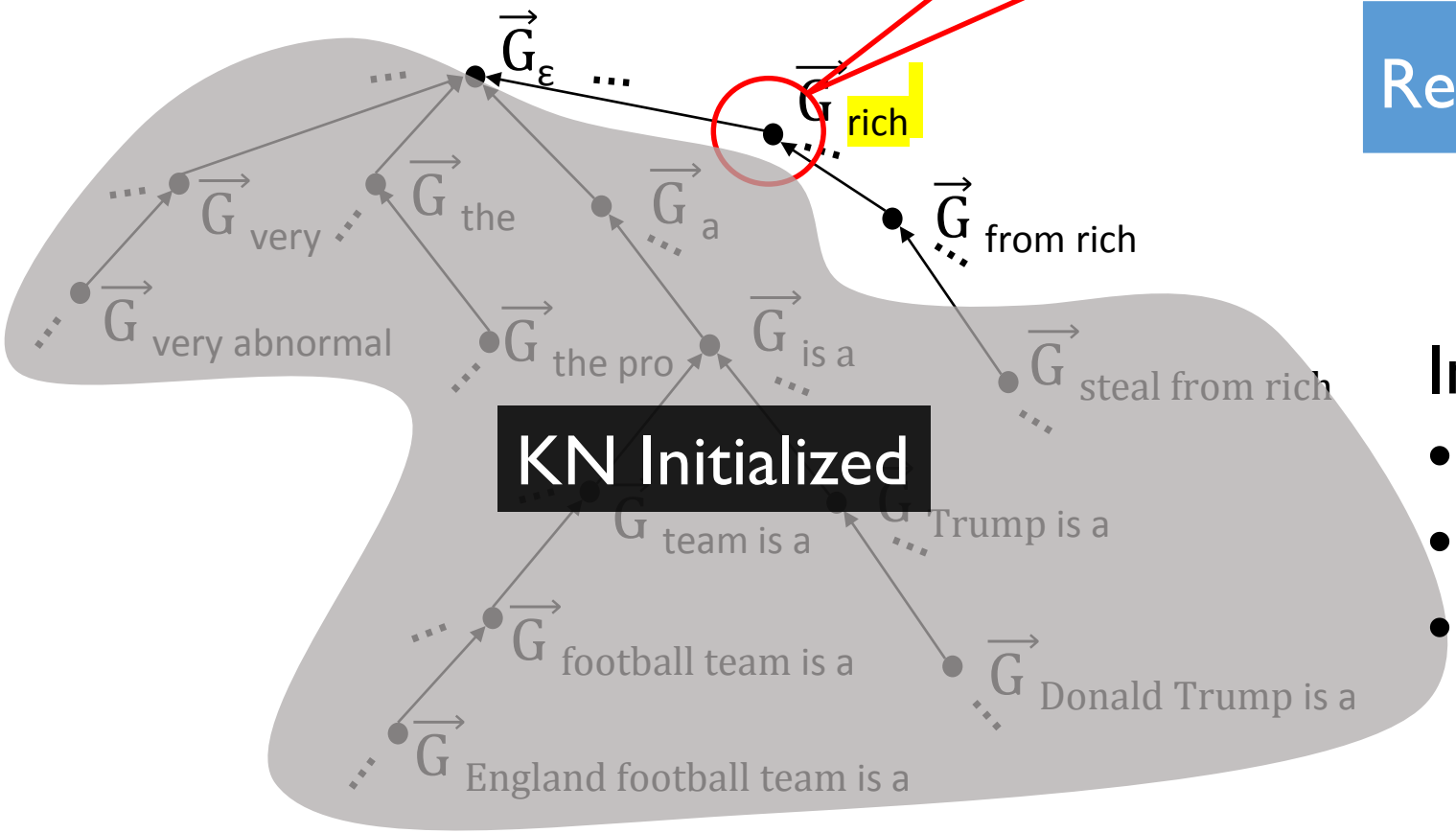


Read $n_{\text{rich}}^{\text{give}}$ from proxy counts

and so on

Sampling $\{n_u^w, t_u^w\}_{w \in u}$

Given a query P(give | from rich)



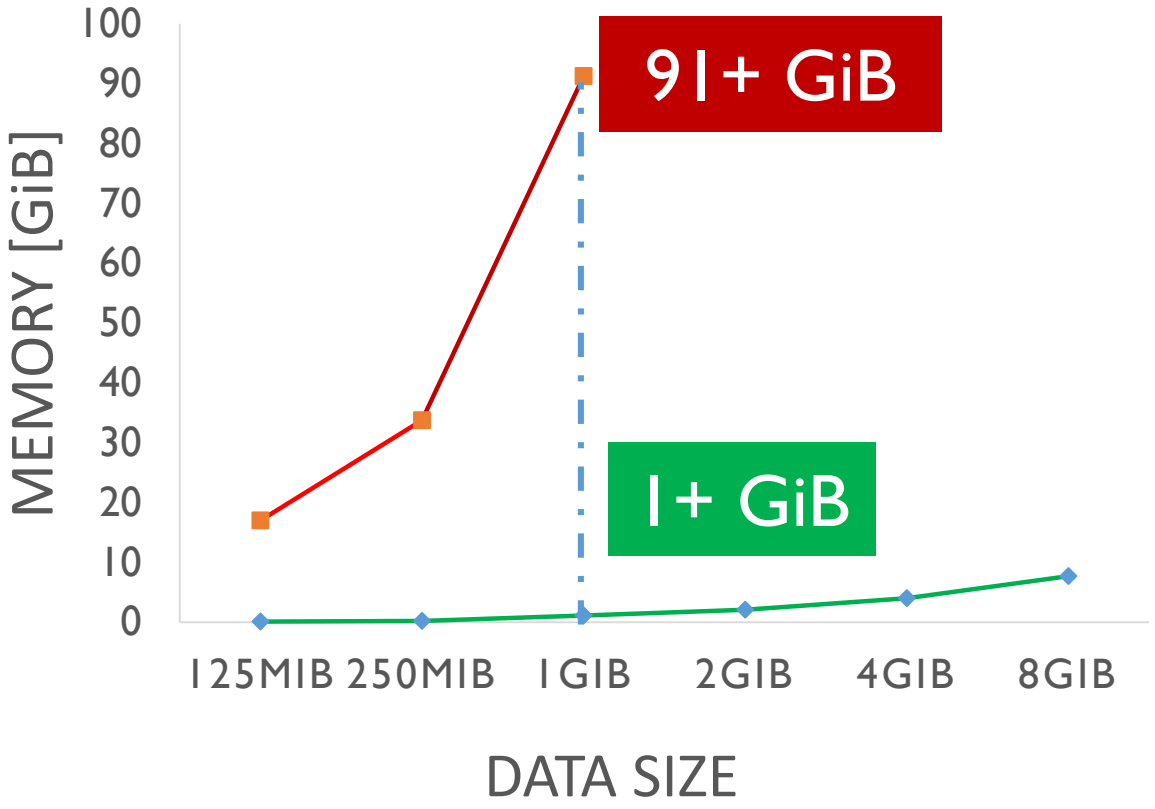
Read $n_{\text{rich}}^{\text{give}}$ from proxy counts

and so on

In practice:

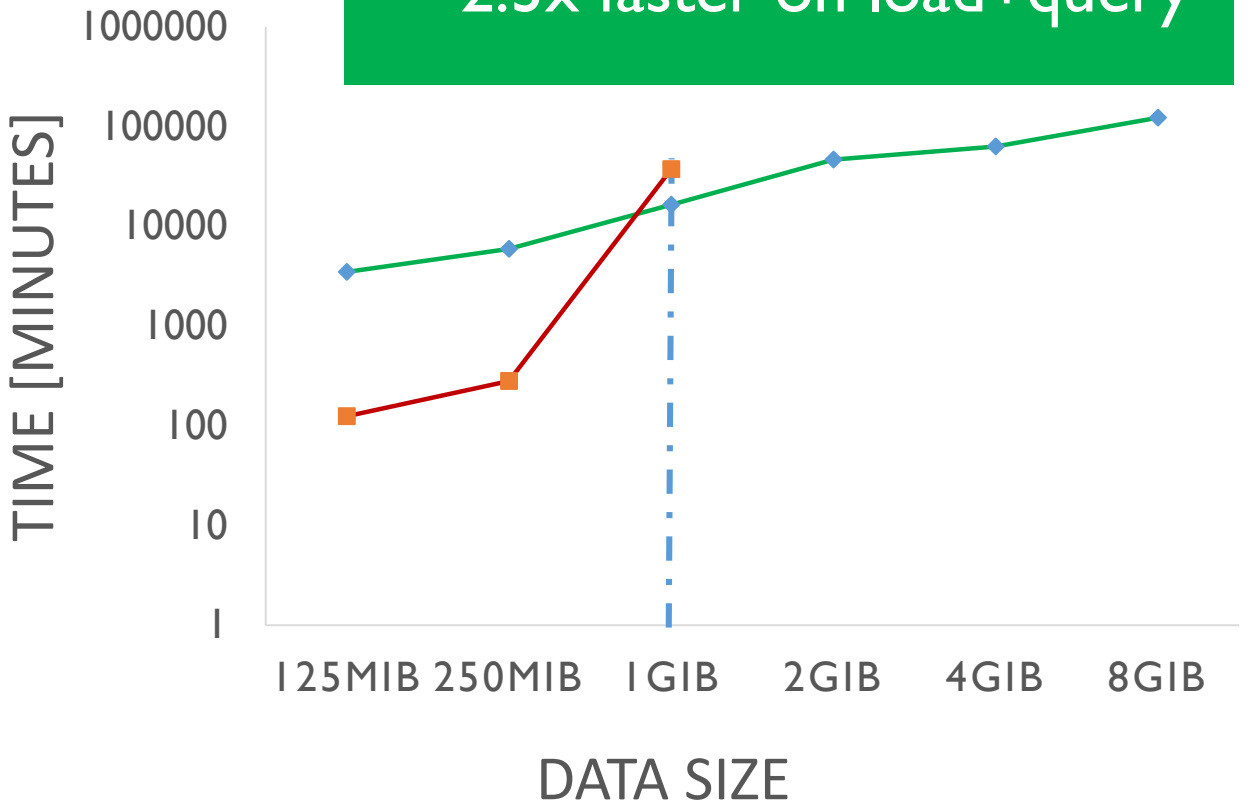
- $0 \leq t_u^w \leq \text{Min}(n_u^w, M)$
- Generate 100 samples per node
- Forget samples

Test time comparison



On 1 GiB:

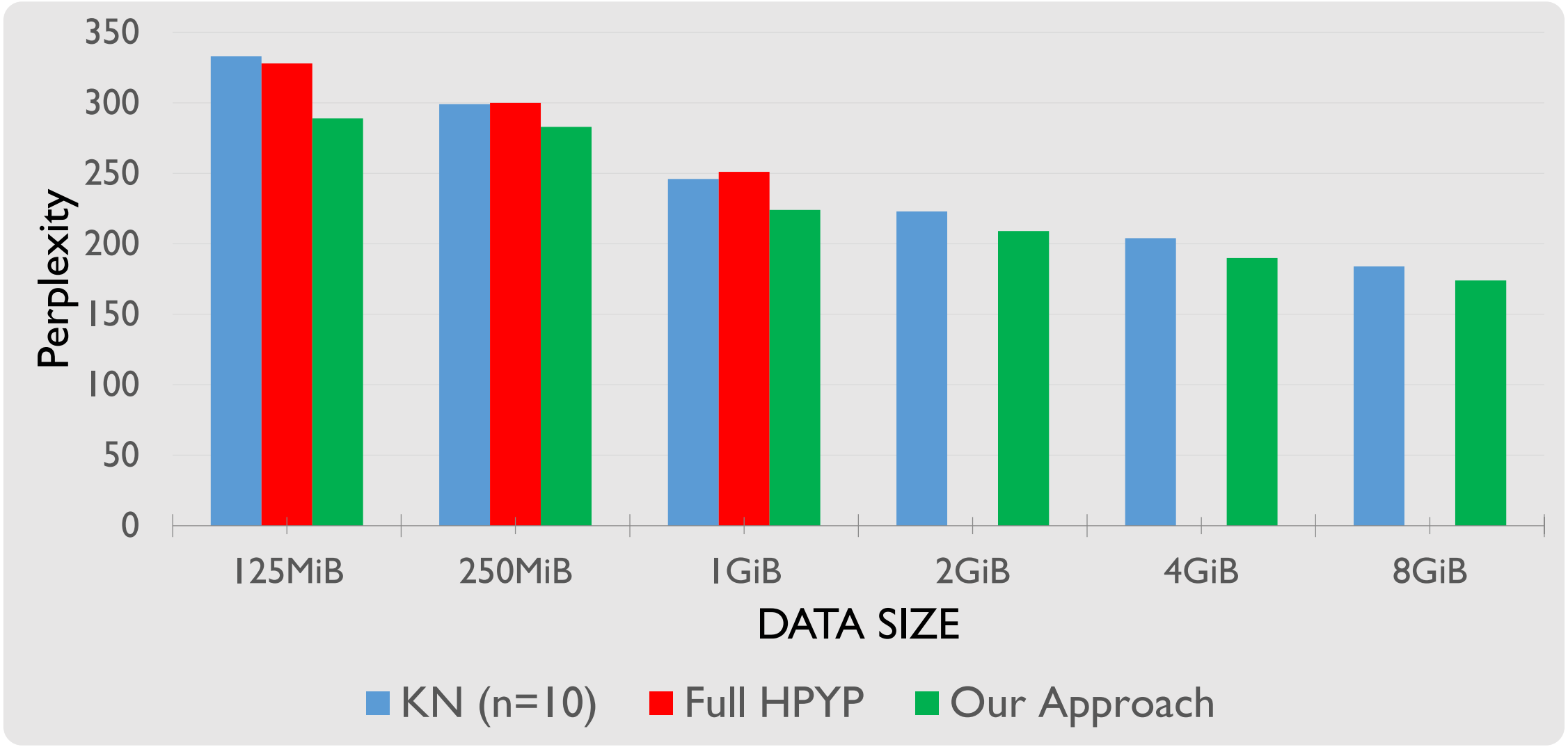
- 1.8x slower on query
- 2.3x faster on load+query



Outline

- Infinite-Order Language Modelling and Challenges
- Compressed HPYP LM
- Inference and Sampling in Compressed HPYP LM
- Perplexity and Mixing
- Conclusion and Future Directions

Perplexity (and Mixing)



Outline

- Infinite-Order Language Modelling and Challenges
- Compressed HPYP LM
- Inference and Sampling in Compressed HPYP LM
- Perplexity and Mixing
- Conclusion and Future Directions

Conclusion

- Proposed a Compressed HPYP LM and a fast and memory-efficient approximate inference scheme.
- Proposed approach is several orders of magnitude smaller than the existing models.
- Avoided potential mixing issues, while consistently outperforming the state-of-the-art count-based language models by a significant margin.

Conclusion

- Proposed a Compressed HPYP LM and a fast and memory-efficient approximate inference scheme.
- Proposed approach is several orders of magnitude smaller than the existing models.
- Avoided potential mixing issues, while consistently outperforming the state-of-the-art count-based language models by a significant margin.

Future Directions

- Sampling speedup (i.e., learning an approximation for Stirling numbers)
- Exploring continuous space approximations of HPYP
- Exploring other applications

Conclusion

- Proposed a Compressed HPYP LM and a fast and memory-efficient approximate inference scheme.
- Proposed approach is several orders of magnitude smaller than the existing models.
- Avoided potential mixing issues, while consistently outperforming the state-of-the-art count-based language models by a significant margin.

Thanks!

Compressed Nonparametric Language Modelling

Slides, supplementary materials, more results available on : eehsan.github.io

Contact: Ehsan.Shareghi@gmail.com