Generating Sentences by Editing Prototypes

Kelvin Guu*, Tatsunori Hashimoto*, Yonatan Oren, Percy Liang

TACL 2018, appeared at ACL 2018

\begin{**Overview**}

Goal: sentence generation

Goal: sentence generation

$p(y) \longrightarrow y =$ "stocks fell by 2 percent"

Goal: sentence generation

$p(y) \longrightarrow$ y = "stocks fell by 2 percent"

x = "死马当活马医" $\xrightarrow{p(y \mid x)}$ y = "beating a dead horse"

Goal: sentence generation

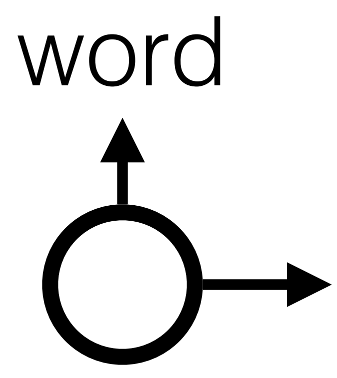
$p(y) \longrightarrow$ y = "stocks fell by 2 percent"

x = "死马当活马医" $\xrightarrow{p(y \mid x)}$ y = "beating a dead horse"

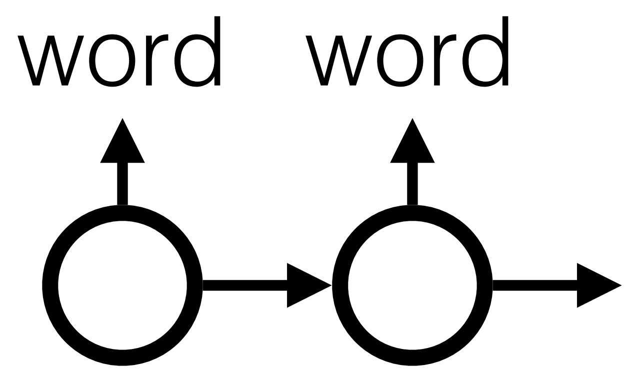
x = "how are you?" $\xrightarrow{p(y \mid x)}$ y = "pretty good, you?"

The status quo

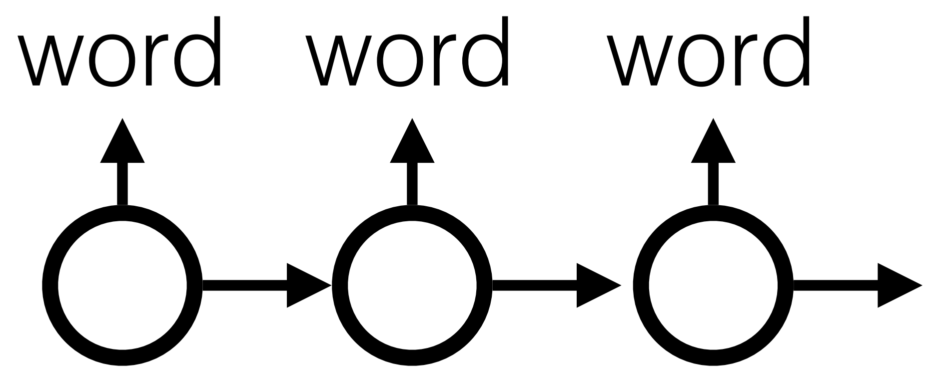
The status quo



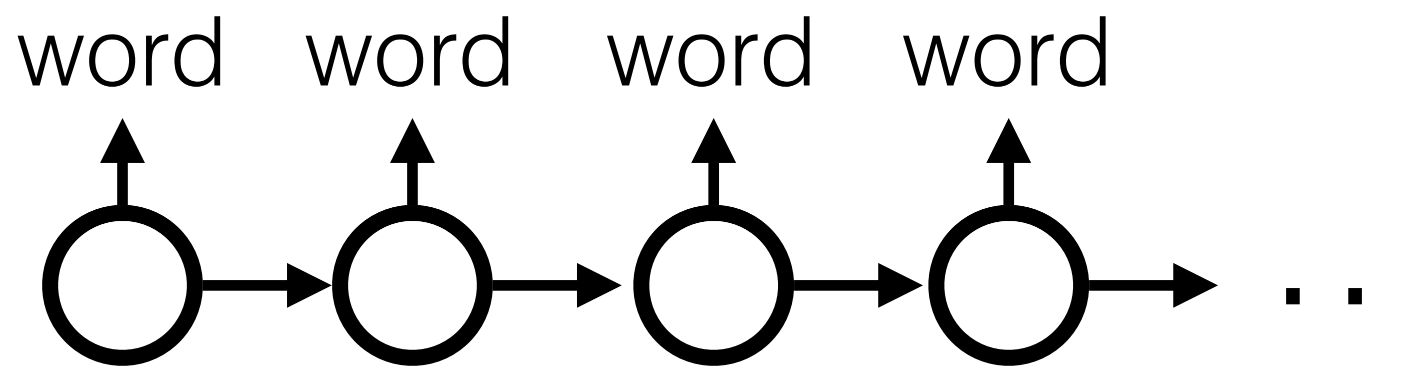
The status quo



The status quo

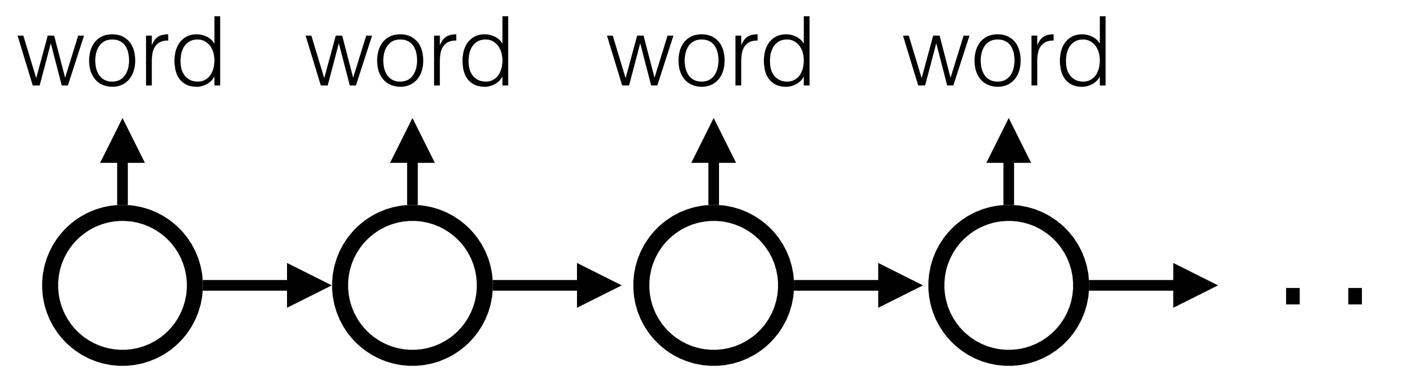


The status quo



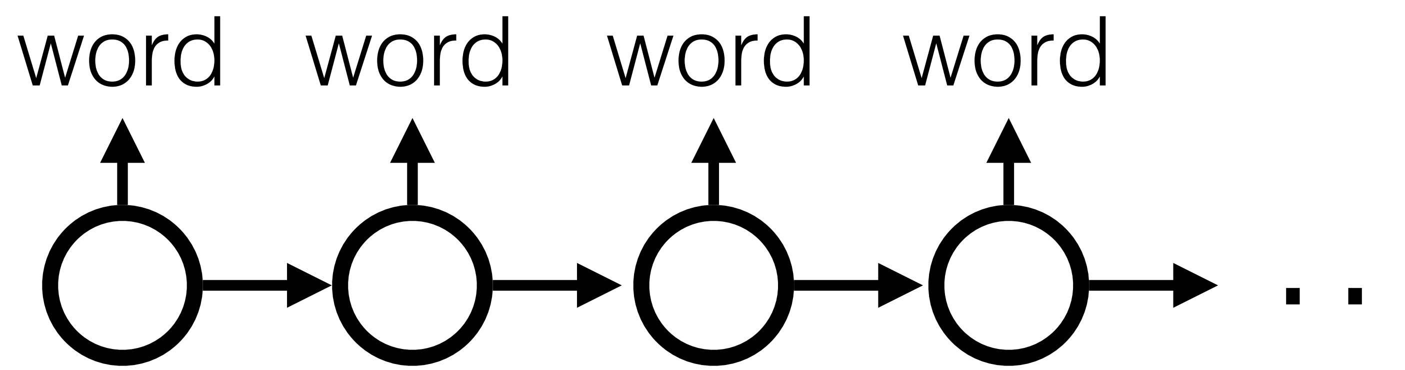
The status quo

- left to right
- word by word



The status quo

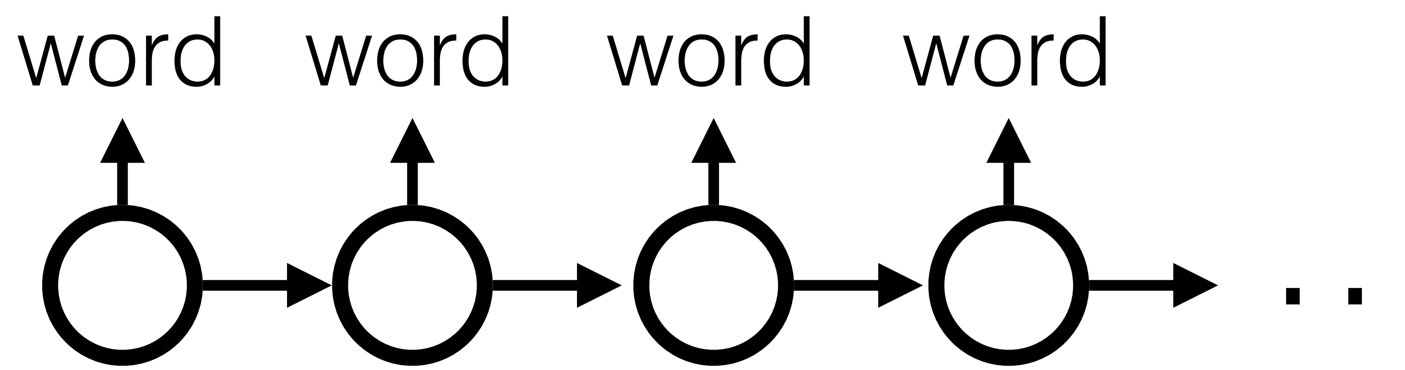
- left to right
- word by word



Train on wide output distributions

The status quo

- left to right
- word by word

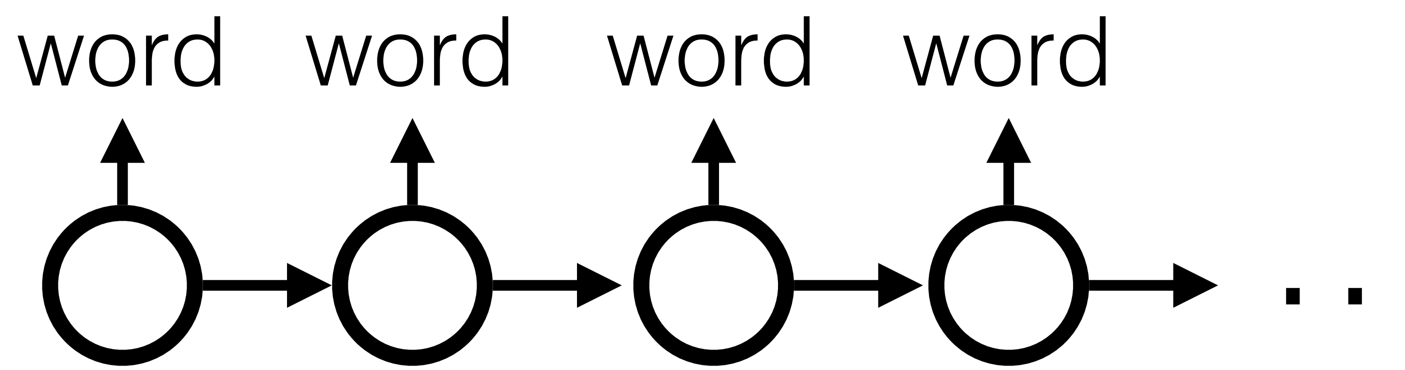


Train on wide output distributions

- low diversity

The status quo

- left to right
- word by word

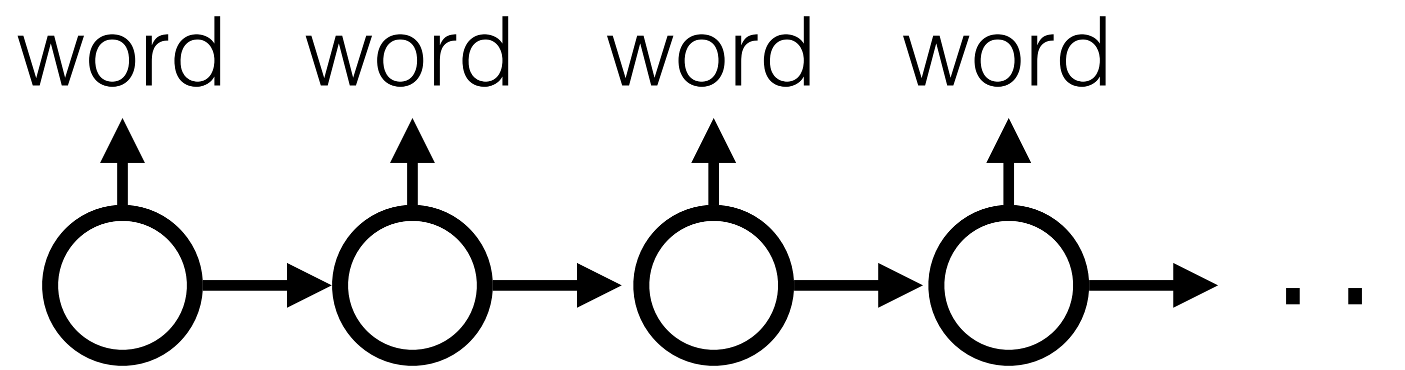


Train on wide output distributions

- low diversity
    - the generic utterance problem

The status quo

- left to right
- word by word

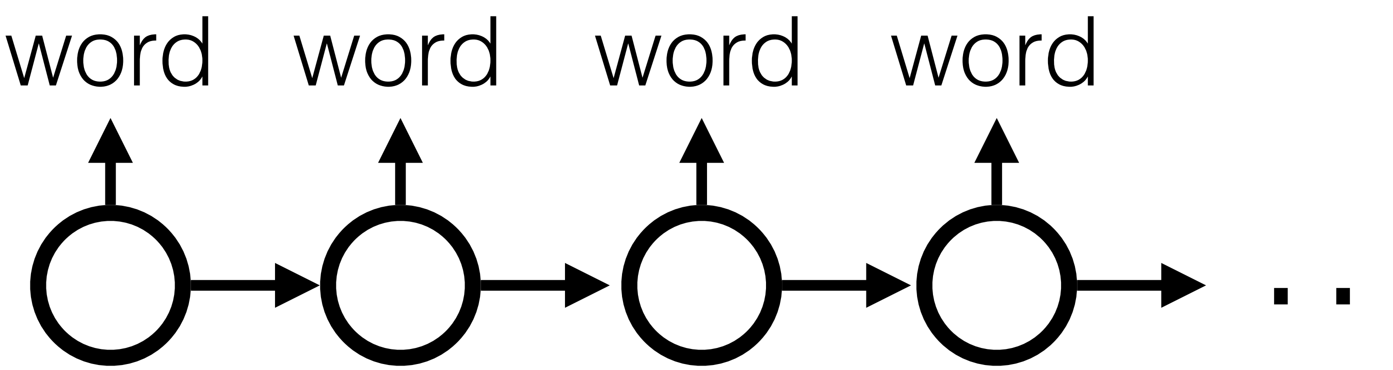


Train on wide output distributions

- low diversity
  - the generic utterance problem
  - (*"I don't know", "I'm sorry"*) [Li+ 2016, Serban+ 2016, Ott+ 2018]

The status quo

- left to right
- word by word



Train on wide output distributions

- low diversity
    - the generic utterance problem
    - (*"I don't know", "I'm sorry"*) [Li+ 2016, Serban+ 2016, Ott+ 2018]
- no semantic control [Hu+ 2017]

Approach: prototype, then edit

Approach: prototype, then edit

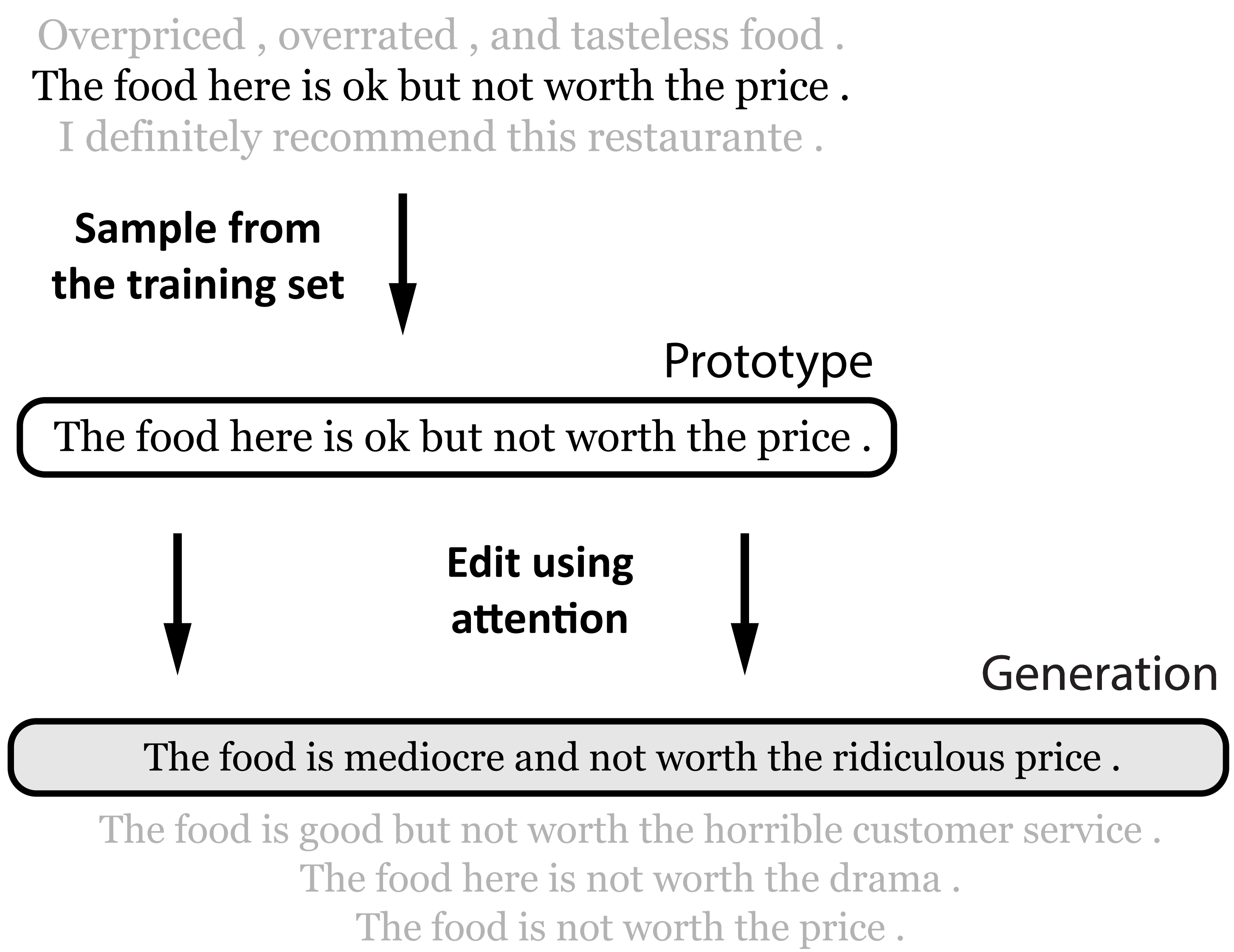
Overpriced , overrated , and tasteless food .
The food here is ok but not worth the price .
I definitely recommend this restaurante .

Sample from the training set

Prototype

The food here is ok but not worth the price .

Approach: prototype, then edit



Approach: prototype, then edit

Overpriced , overrated , and tasteless food .
The food here is ok but not worth the price .
I definitely recommend this restaurante .

Sample from the training set

Prototype

The food here is ok but not worth the price .

prototype controls rough semantics

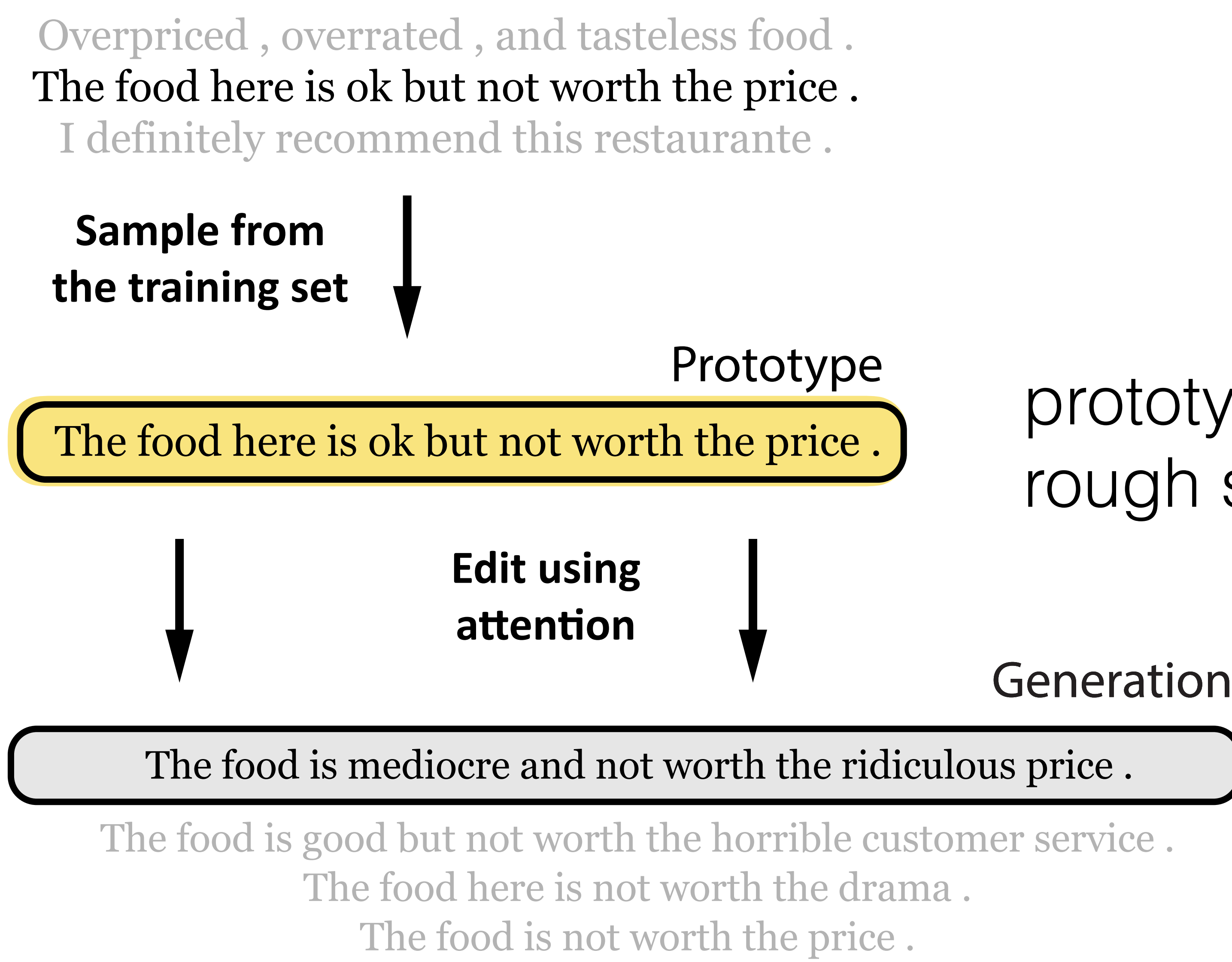
Edit using attention

Generation

The food is mediocre and not worth the ridiculous price .

The food is good but not worth the horrible customer service .
The food here is not worth the drama .
The food is not worth the price .

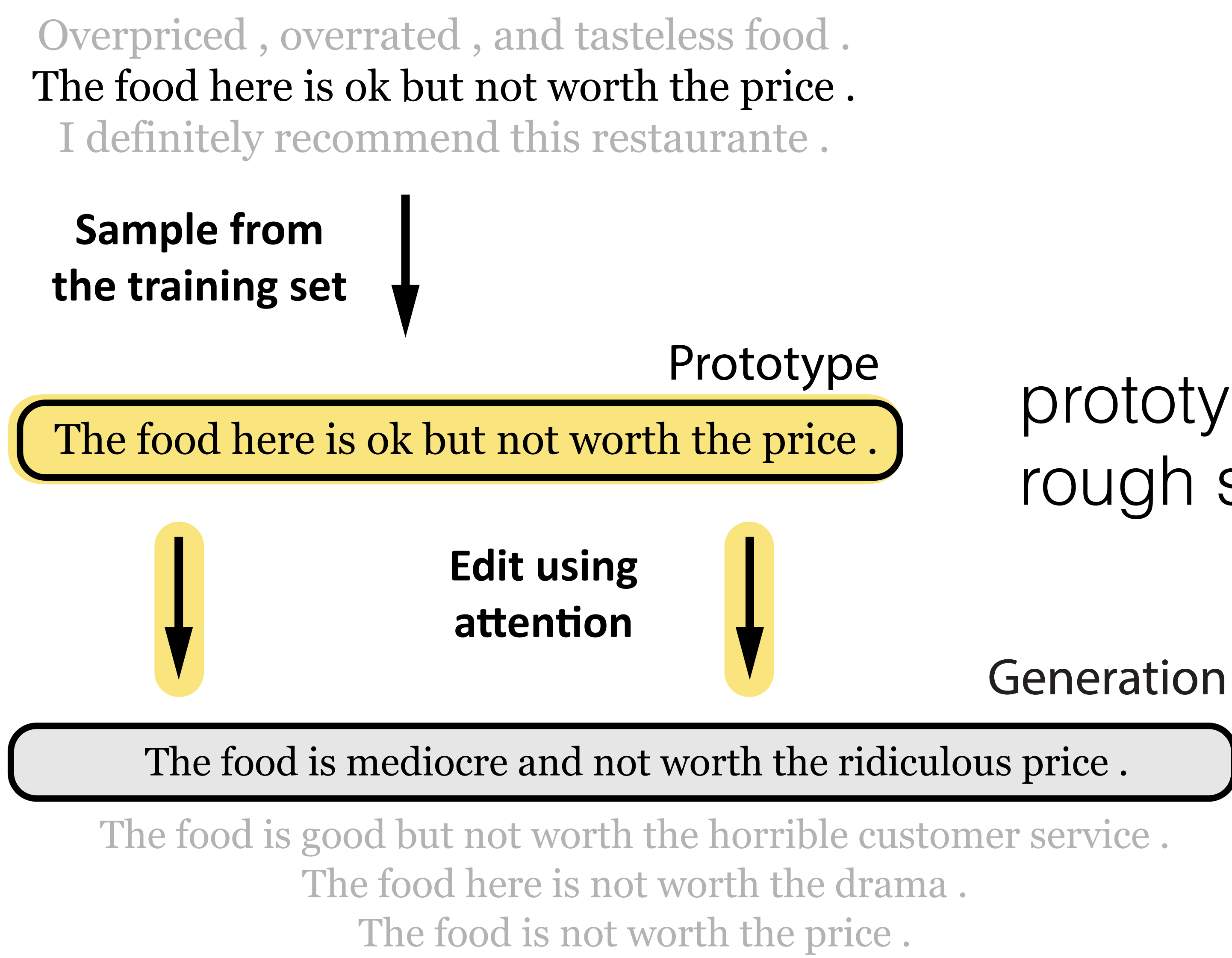
Approach: prototype, then edit



prototype controls rough semantics

guarantee diversity

Approach: prototype, then edit



prototype controls rough semantics

guarantee diversity

seq2seq editor injects variation

Overview of results

Overview of results

- **More diverse generations**

Overview of results

- **More diverse generations**
- **Higher quality generations** (Mechanical Turk)

Overview of results

- **More diverse generations**
- **Higher quality generations** (Mechanical Turk)
- **Better perplexity** (BillionWord, Yelp reviews)

Overview of results

- **More diverse generations**
- **Higher quality generations** (Mechanical Turk)
- **Better perplexity** (BillionWord, Yelp reviews)
- **Seq2seq edits are semantically interpretable**

Overview of results

- **More diverse generations**
- **Higher quality generations** (Mechanical Turk)
- **Better perplexity** (BillionWord, Yelp reviews)
- **Seq2seq edits are semantically interpretable**
  - preserve semantic similarity

Overview of results

- **More diverse generations**
- **Higher quality generations** (Mechanical Turk)
- **Better perplexity** (BillionWord, Yelp reviews)
- **Seq2seq edits are semantically interpretable**
  - preserve semantic similarity
  - can be used to perform sentence-level analogies

\end{**Overview**}

\begin{**Approach**}

prototype, then edit **(formally)**

prototype, then edit **(formally)**

Overpriced , overrated , and tasteless food .
The food here is ok but not worth the price .
I definitely recommend this restaurante .

Sample from the training set

Prototype

The food here is ok but not worth the price .

prototype, then edit **(formally)**

Overpriced , overrated , and tasteless food .
The food here is ok but not worth the price .
I definitely recommend this restaurante .

Sample from the training set

Prototype

The food here is ok but not worth the price .

$$z_p \sim p_{\text{proto}}$$

prototype, then edit **(formally)**

Overpriced , overrated , and tasteless food .
The food here is ok but not worth the price .
I definitely recommend this restaurante .

Sample from the training set

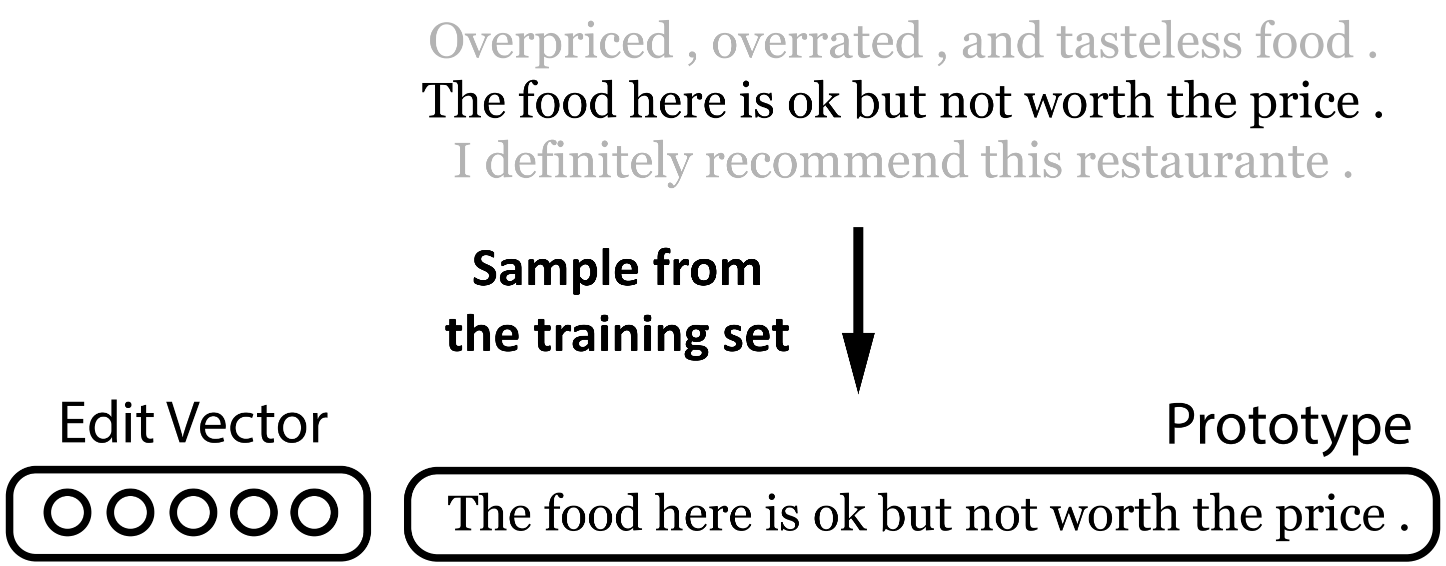
Edit Vector

Prototype

The food here is ok but not worth the price .

$$z_p \sim p_{\text{proto}}$$

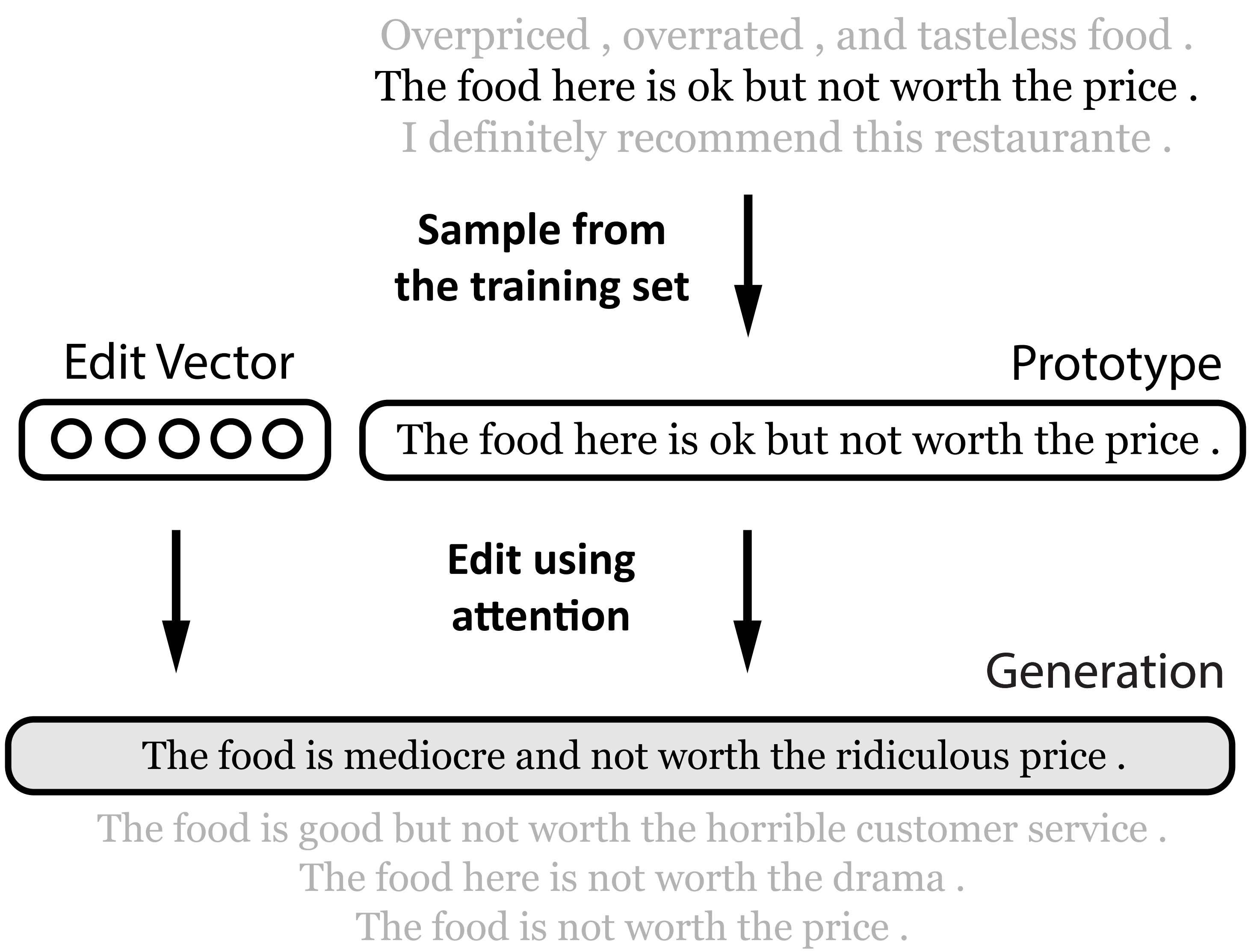
prototype, then edit **(formally)**



$$z_p \sim p_{\text{proto}}$$

$$z_e \sim p_{\text{edit}}$$

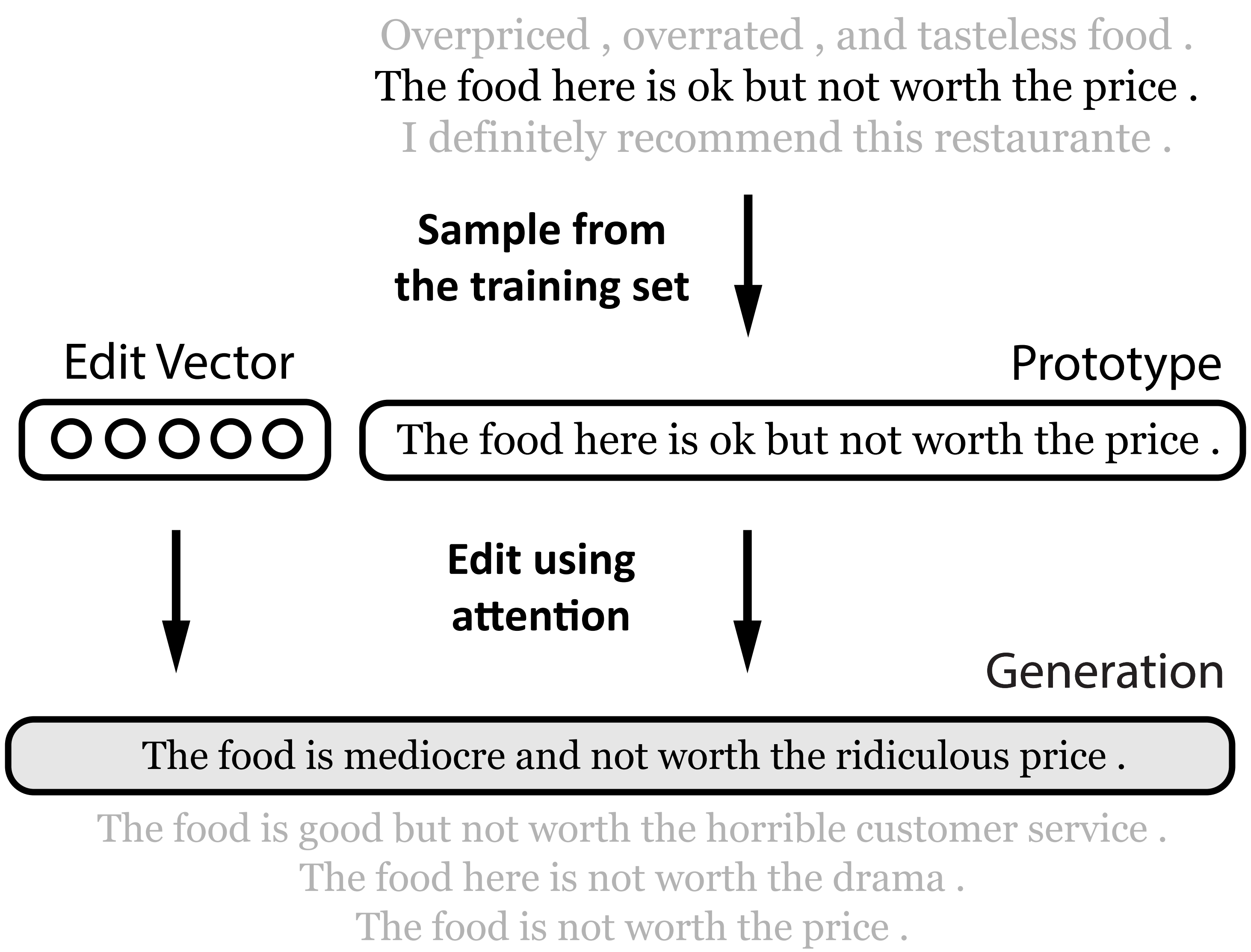
prototype, then edit **(formally)**



$$z_p \sim p_{\text{proto}}$$

$$z_e \sim p_{\text{edit}}$$

prototype, then edit **(formally)**

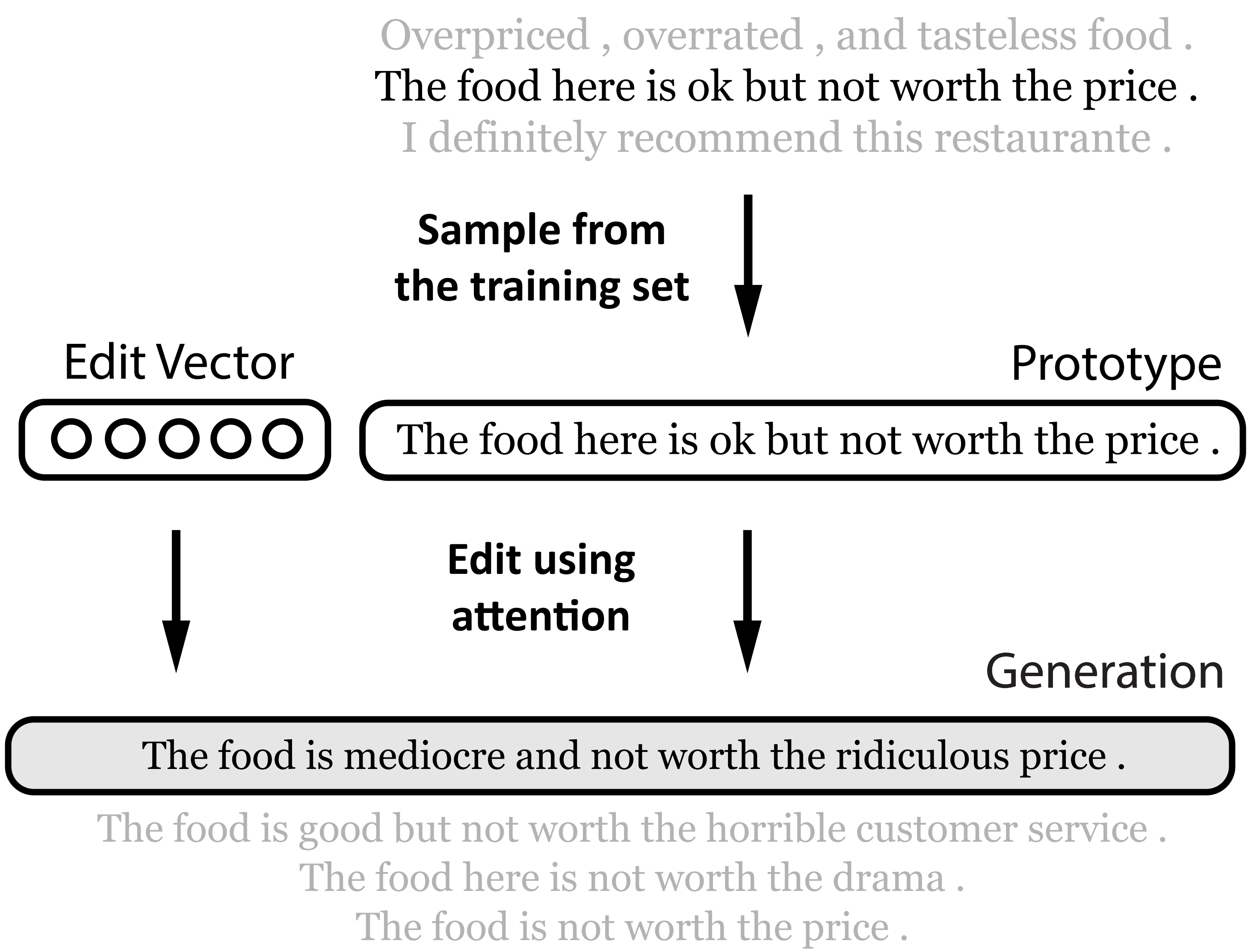


$$z_p \sim p_{\text{proto}}$$

$$z_e \sim p_{\text{edit}}$$

$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

prototype, then edit **(formally)**



$$z_p \sim p_{\text{proto}}$$

$$z_e \sim p_{\text{edit}}$$

$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Intuitions

Intuitions

humans are not pure left-to-right generators

Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit

Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates

Intuitions

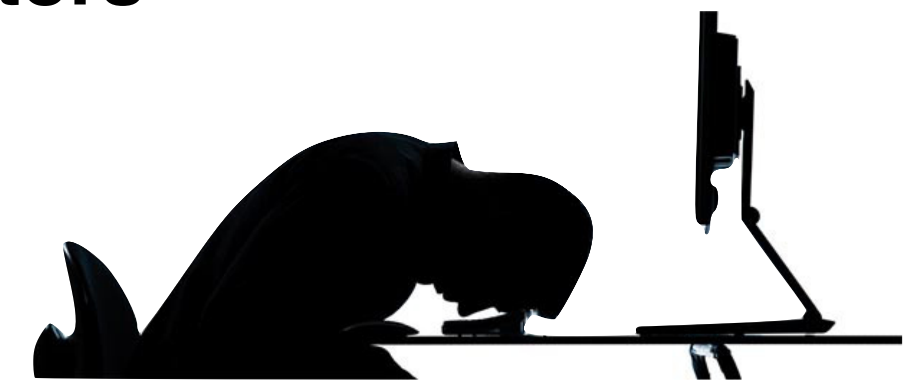
humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize

Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize



Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize

semi-parametric statistics

Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize

semi-parametric statistics

- we are doing **kernel density estimation** over sentence space

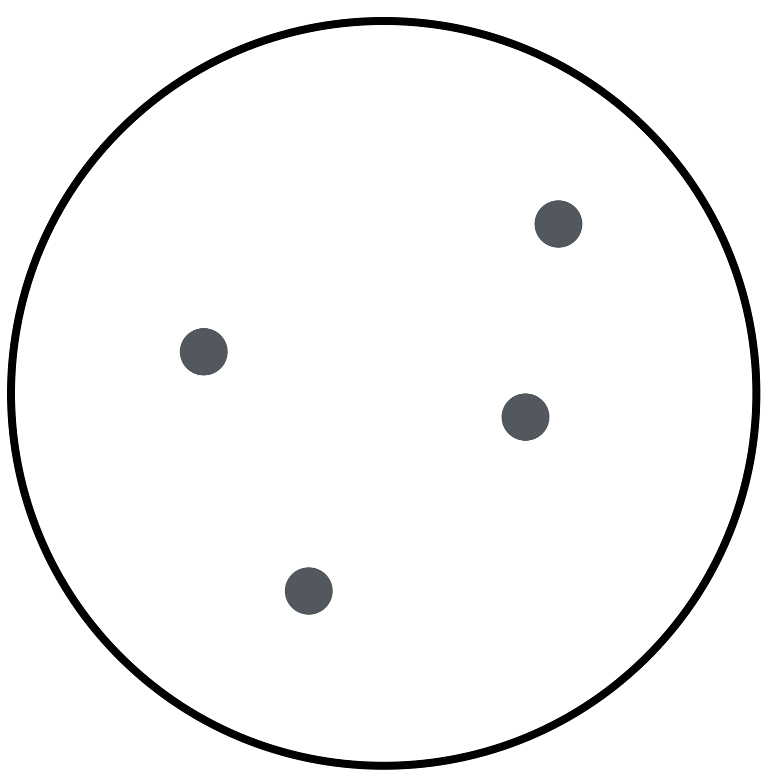
Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize

semi-parametric statistics

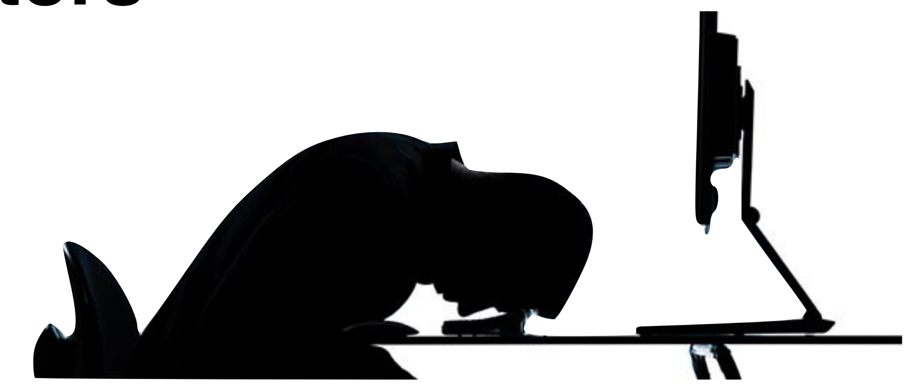
- we are doing **kernel density estimation** over sentence space



Intuitions

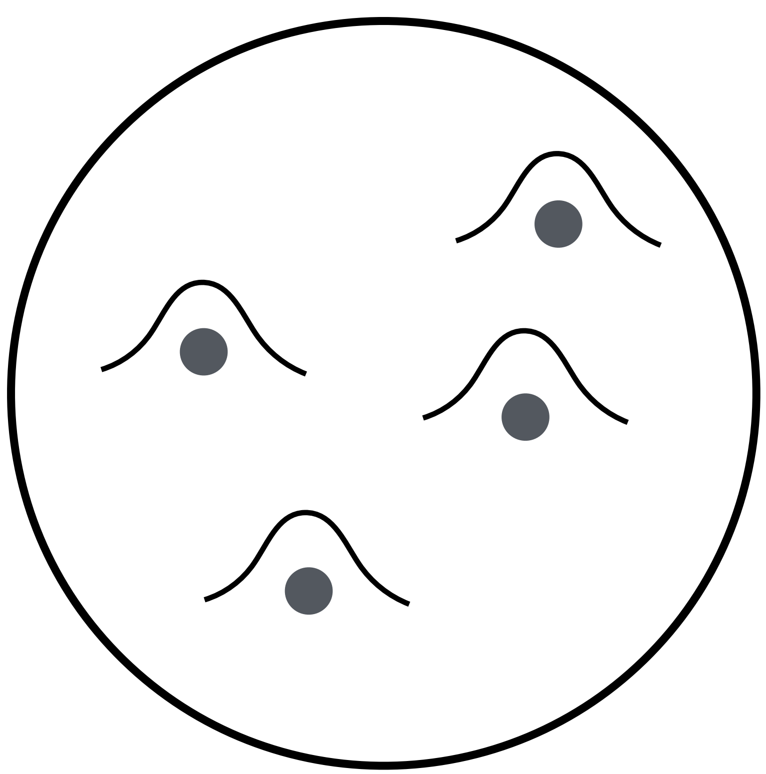
humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize



semi-parametric statistics

- we are doing **kernel density estimation** over sentence space



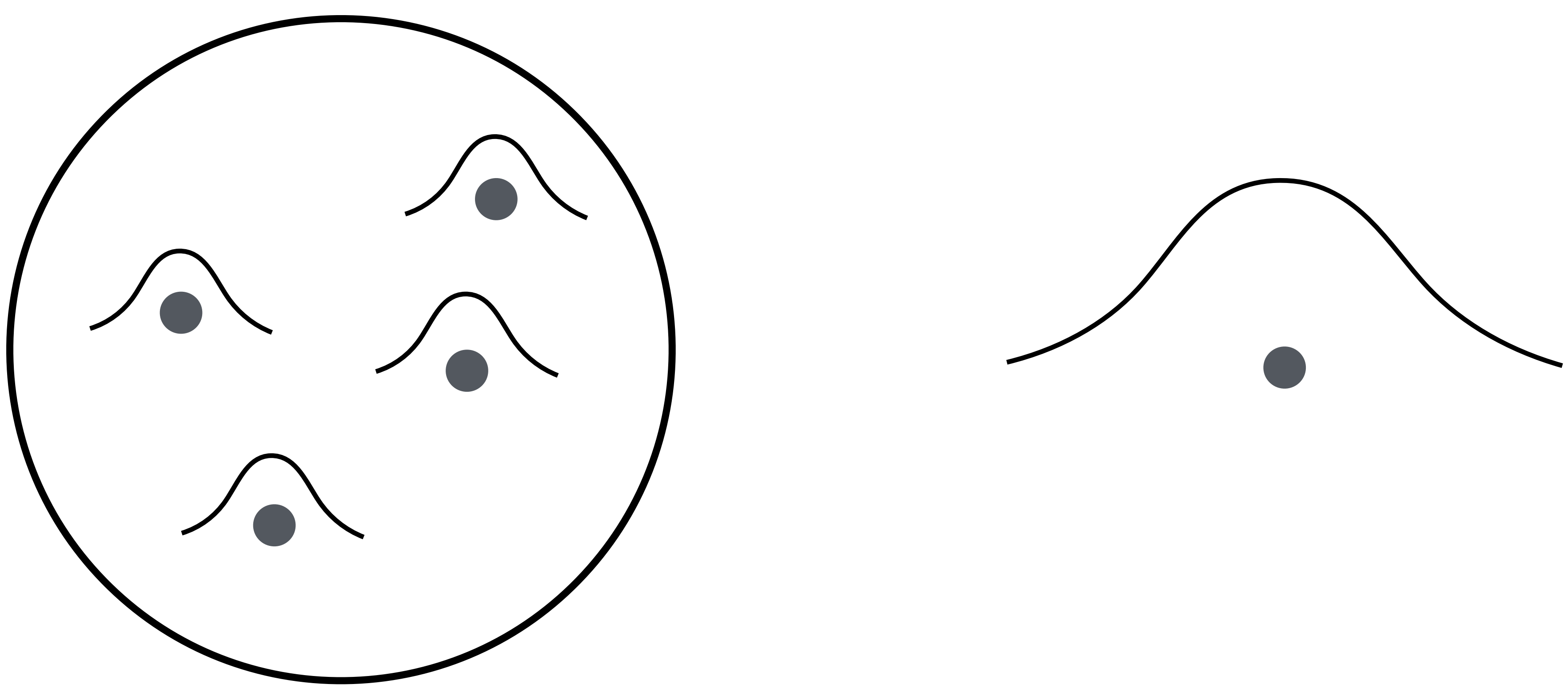
Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize

semi-parametric statistics

- we are doing **kernel density estimation** over sentence space



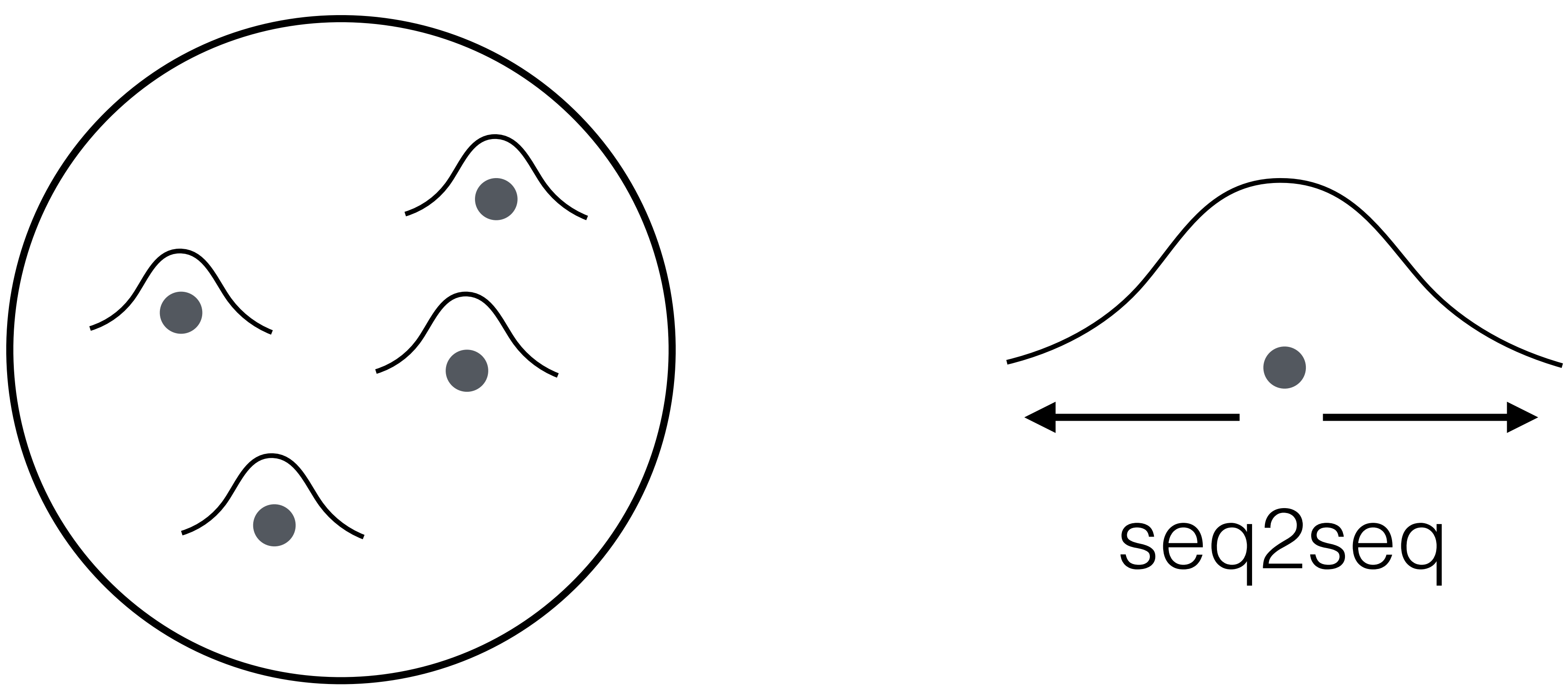
Intuitions

humans are not pure left-to-right generators

- we write a first draft, then edit
- we use templates
- we plagiarize

semi-parametric statistics

- we are doing **kernel density estimation** over sentence space



Another intuition

Another intuition

Professor of Computer Science
The University of Texas at Austin

Another intuition

Professor of Computer Science
The University of Texas at Austin

*You can't cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!*

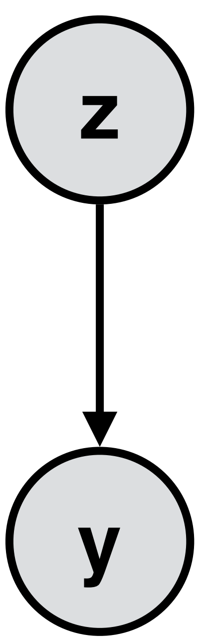
[Ray Mooney, ACL 2014]

Another intuition

Professor of Computer Science
The University of Texas at Austin

*You can't cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!*

[Ray Mooney, ACL 2014]



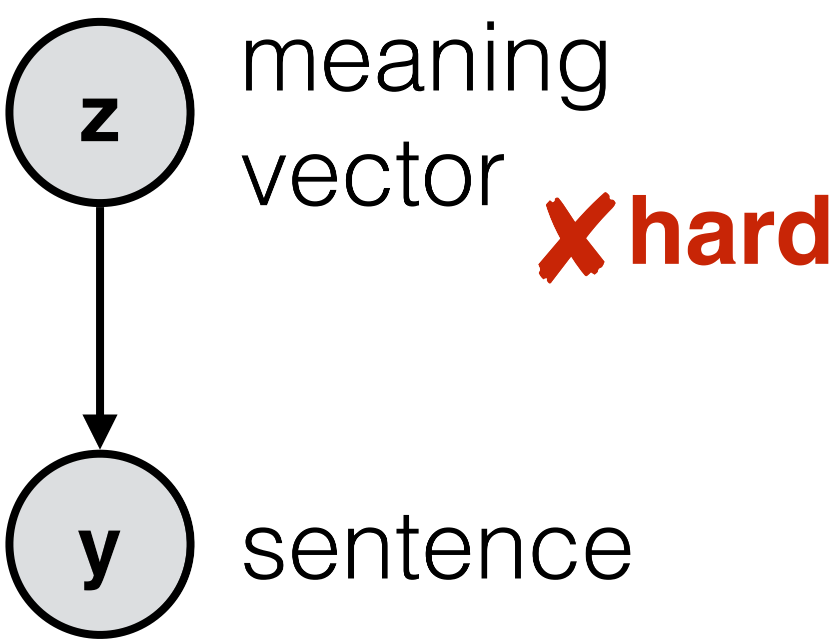
meaning vector

sentence

Another intuition

Professor of Computer Science
The University of Texas at Austin

> *You can't cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!*
>
> [Ray Mooney, ACL 2014]

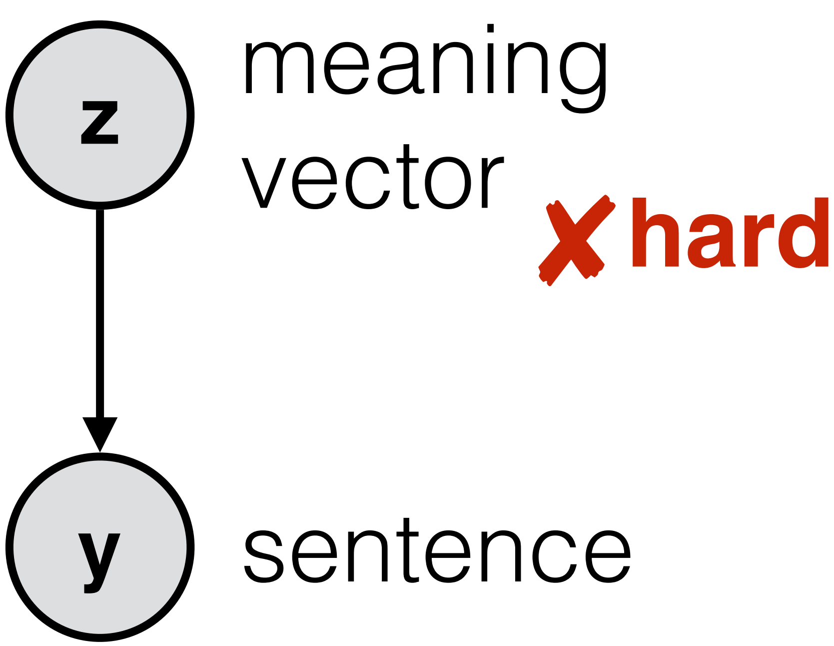


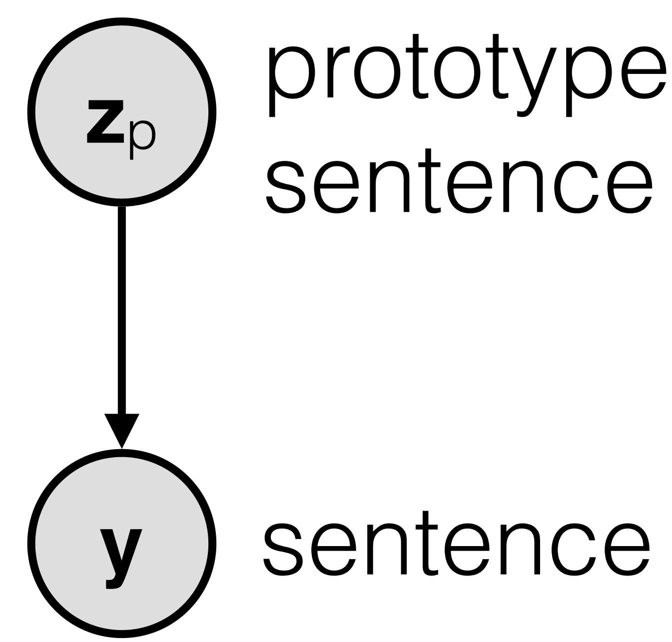
Another intuition

Professor of Computer Science
The University of Texas at Austin

*You can't cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!*

[Ray Mooney, ACL 2014]



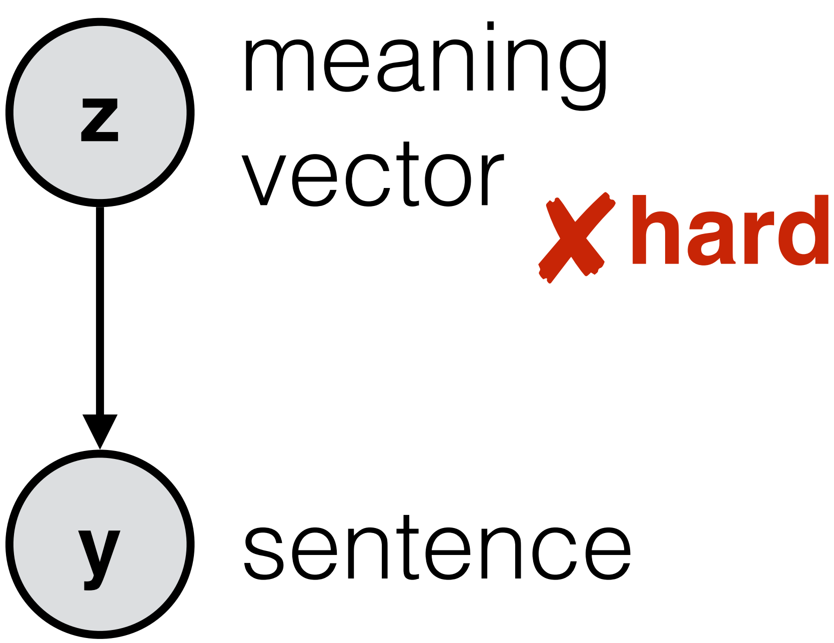


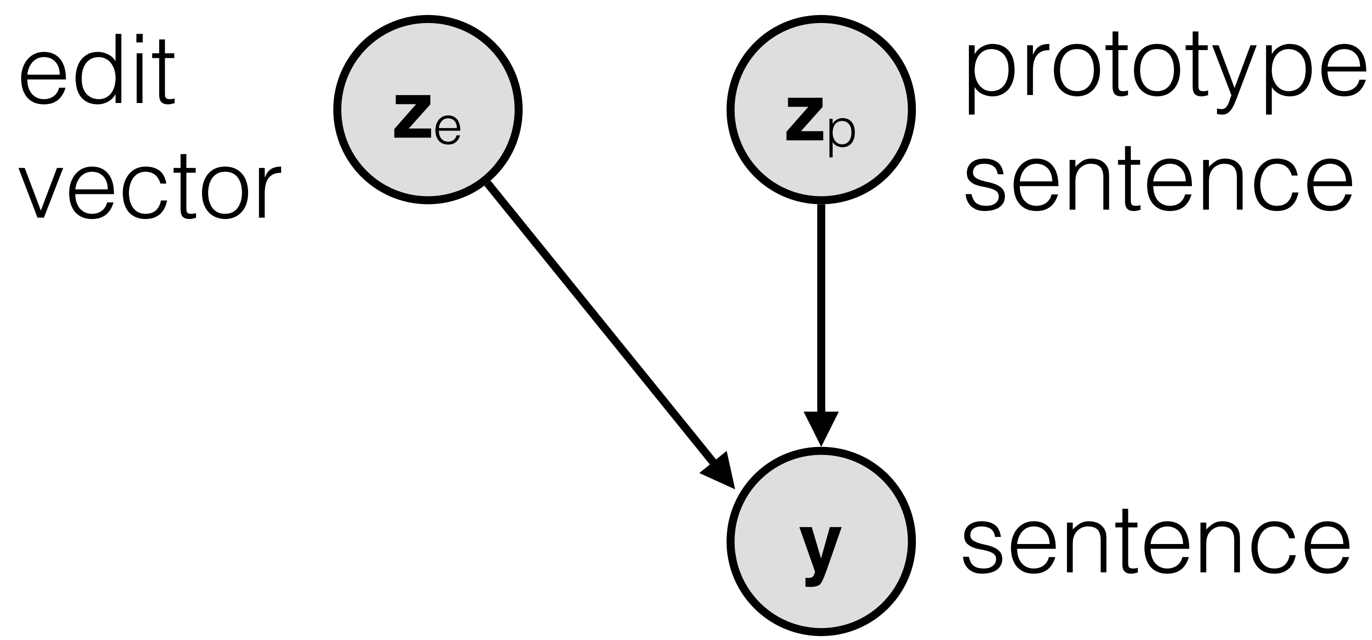
Another intuition

Professor of Computer Science
The University of Texas at Austin

*You can't cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!*

[Ray Mooney, ACL 2014]



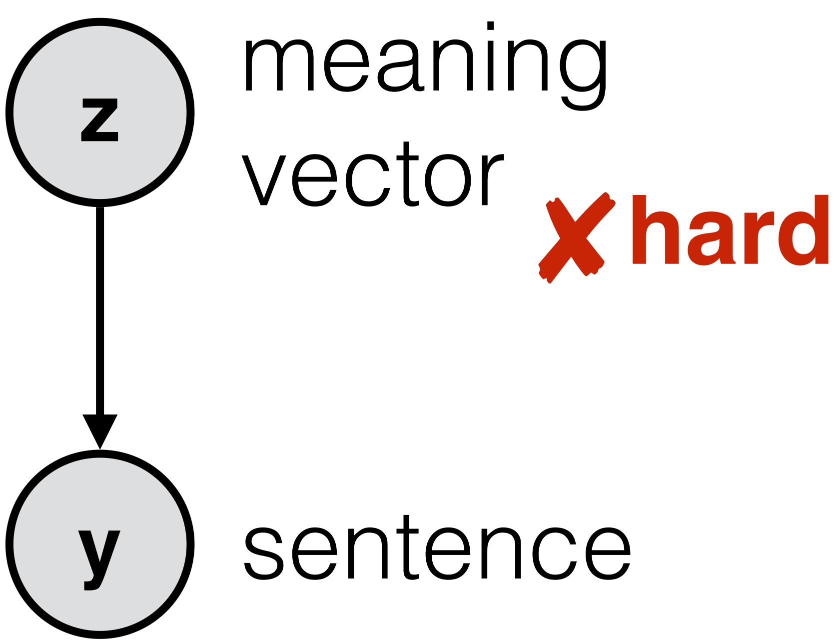


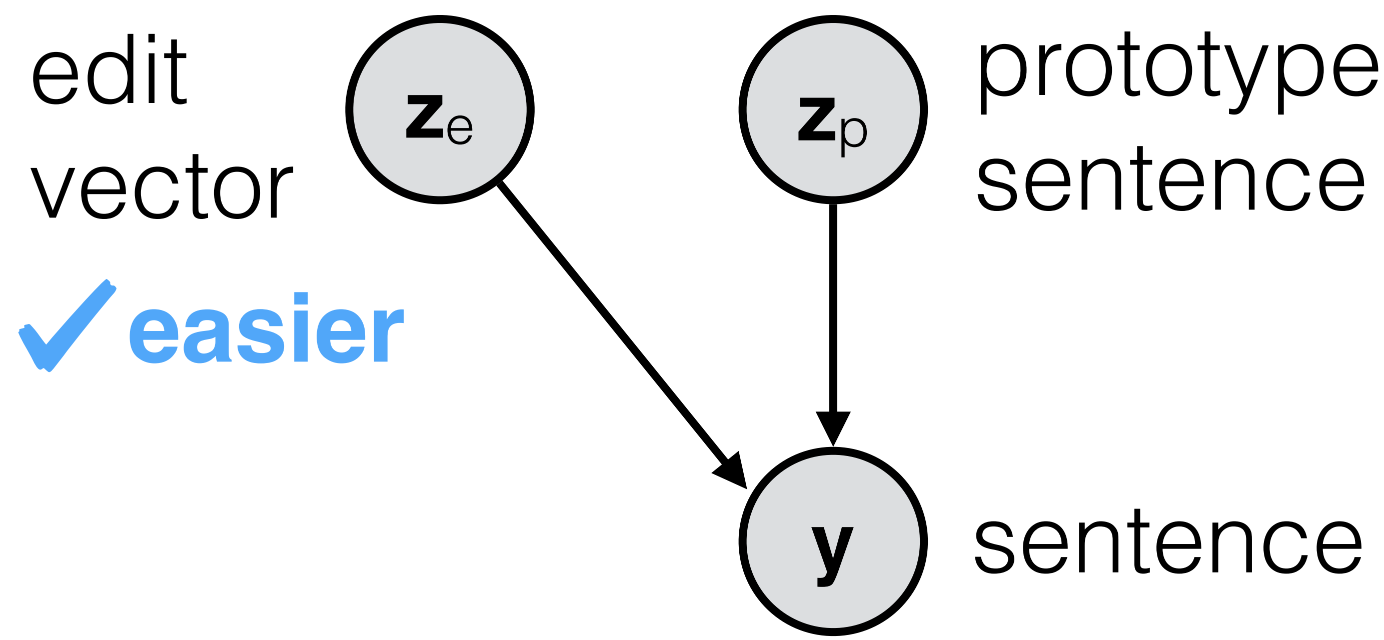
Another intuition

Professor of Computer Science
The University of Texas at Austin

*You can't cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!*

[Ray Mooney, ACL 2014]





Training objective

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Training objective

$$p(y)$$

y = output sentence **z**$_{\text{p}}$ = prototype sentence **z**$_{\text{e}}$ = edit vector

Training objective

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Training objective

$$p\left(y\right) = \sum_{z_p} \underbrace{p\left(y \mid z_p\right)}_{\int_{z_e} p_{\text{editor}}\left(y \mid z_p, z_e\right) p_{\text{edit}}\left(z_e\right) dz_e} p_{\text{proto}}\left(z_p\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Training objective

maximize

$$p(y) = \sum_{z_p} \underbrace{p(y \mid z_p)}_{\int_{z_e} p_{\text{editor}}(y \mid z_p, z_e)\, p_{\text{edit}}(z_e)\, dz_e} p_{\text{proto}}(z_p)$$

y = output sentence **z**p = prototype sentence **z**e = edit vector

Training objective

maximize

$$p\left(y\right) = \sum_{z_p} \underbrace{p\left(y \mid z_p\right)}_{\int_{z_e} p_{\text{editor}}\left(y \mid z_p, z_e\right) p_{\text{edit}}\left(z_e\right) dz_e} p_{\text{proto}}\left(z_p\right)$$

expensive

y = output sentence **z**p = prototype sentence **z**e = edit vector

Training objective

maximize

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

expensive

$$\int_{z_e} p_{\text{editor}}(y \mid z_p, z_e)\, p_{\text{edit}}(z_e)\, dz_e$$

intractable

y = output sentence **z**p = prototype sentence **z**e = edit vector

Training objective

maximize

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

expensive

$$\int_{z_e} p_{\text{editor}}(y \mid z_p, z_e)\, p_{\text{edit}}(z_e)\, dz_e$$

intractable

key tool: **ELBO** (evidence lower bound)

[Dempster+ '77, Jordan+ '99, Kingma+ '13]

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Training objective

maximize

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$ expensive

$$\int_{z_e} p_{\text{editor}}(y \mid z_p, z_e)\, p_{\text{edit}}(z_e)\, dz_e$$ intractable

key tool: **ELBO** (evidence lower bound)

[Dempster+ '77, Jordan+ '99, Kingma+ '13]

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Training objective

maximize

ELBO

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

expensive

ELBO

$$\int_{z_e} p_{\text{editor}}(y \mid z_p, z_e)\, p_{\text{edit}}(z_e)\, dz_e$$

intractable

key tool: **ELBO** (evidence lower bound)

[Dempster+ '77, Jordan+ '99, Kingma+ '13]

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Training objective

maximize

ELBO

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

expensive

ELBO

$$\int_{z_e} p_{\text{editor}}(y \mid z_p, z_e)\, p_{\text{edit}}(z_e)\, dz_e$$

intractable

key tool: **ELBO** (evidence lower bound)

[Dempster+ '77, Jordan+ '99, Kingma+ '13]

- more computationally tractable

y = output sentence **z_p** = prototype sentence **z_e** = edit vector

Training objective

maximize

ELBO

$$p\left(y\right) = \sum_{z_p} p\left(y \mid z_p\right) p_{\text{proto}}\left(z_p\right)$$

expensive

ELBO

$$\int_{z_e} p_{\text{editor}}\left(y \mid z_p, z_e\right) p_{\text{edit}}\left(z_e\right) dz_e$$

intractable

key tool: **ELBO** (evidence lower bound)

[Dempster+ '77, Jordan+ '99, Kingma+ '13]

- more computationally tractable
- bias towards semantically interpretable edits

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

ELBO (in general)

$$\log p(y)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\log p\,(y)$$

$$\int_z p\,(y \mid z)\, p\,(z)\, dz$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\log p(y) \quad \geq \quad \int_z \log p(y \mid z)\, q(z)\, dz - KL\left(q(z) \,\|\, p(z)\right)$$

$$\int_z p(y \mid z)\, p(z)\, dz$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\underbrace{\log p(y)}_{\int_z p(y \mid z)\, p(z)\, dz} \geq \int_z \log p(y \mid z)\, \underbrace{q(z)}_{\mathbf{q(z)}}\, dz - KL\left(\underbrace{q(z)}_{\mathbf{q(z)}} \| p(z)\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\underbrace{\log p\left(y\right)}_{\int_z p\left(y \mid z\right) p\left(z\right) dz} \geq \int_z \log p\left(y \mid z\right) \underbrace{q\left(z\right)}_{q(z)} dz - KL\left(\underbrace{q\left(z\right)}_{q(z)} \| p\left(z\right)\right)$$

q(z)

you choose q(z)

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\underbrace{\log p(y)}_{\int_z p(y \mid z)\, p(z)\, dz} \geq \int_z \log p(y \mid z)\, \underbrace{q(z)}_{q(z)}\, dz - KL\left(\underbrace{q(z)}_{q(z)} \,\|\, p(z)\right)$$

you choose q(z)

- add helpful biases to the model

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\underbrace{\log p(y)}_{\int_z p(y \mid z)\, p(z)\, dz} \geq \int_z \log p(y \mid z)\, \underbrace{q(z)}_{q(z)}\, dz - KL\left(\underbrace{q(z)}_{q(z)} \| p(z)\right)$$

you choose q(z)

- add helpful biases to the model
- tightness of the lower bound

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\log \underbrace{p\left(y\right)}_{\int_z p\left(y \mid z\right) p\left(z\right) dz} \geq \int_z \log p\left(y \mid z\right) \underbrace{q\left(z\right)}_{\mathbf{q(z)}} dz - KL\left(\underbrace{q\left(z\right)}_{\mathbf{q(z)}} \| p\left(z\right)\right)$$

you choose q(z)

- add helpful biases to the model
- tightness of the lower bound $\quad q\left(z\right) \approx p\left(z \mid y\right)$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO (in general)

$$\underbrace{\log p(y)}_{\int_z p(y \mid z)\, p(z)\, dz} \geq \int_z \log p(y \mid z)\, \underbrace{q(z)}_{q(z)}\, dz - KL\left(\underbrace{q(z)}_{q(z)} \| p(z)\right)$$

q(z)

you choose q(z)

- add helpful biases to the model
- tightness of the lower bound

$$\boxed{q(z) \approx p(z \mid y)}$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Training objective

maximize

ELBO

$$p\left(y\right) = \sum_{z_p} p\left(y \mid z_p\right) p_{\text{proto}}\left(z_p\right)$$

expensive

ELBO

$$\int_{z_e} p_{\text{editor}}\left(y \mid z_p, z_e\right) p_{\text{edit}}\left(z_e\right) dz_e$$

intractable

y = output sentence **z**p = prototype sentence **z**e = edit vector

ELBO on prototypes

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO on prototypes

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\mathrm{proto}}(z_p)$$

$$\geq \quad \sum_{z_p} \log p(y \mid z_p)\, q(z_p) - KL\left(q(z_p) \,\|\, p_{\mathrm{proto}}(z_p)\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO on prototypes

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

$$\geq \quad \sum_{z_p} \log p(y \mid z_p)\, q(z_p) - KL\left(q(z_p) \,\|p_{\text{proto}}(z_p)\right)$$

$$q(z_p) \approx p(z_p \mid y)$$

y = output sentence **z**p = prototype sentence **z**e = edit vector

ELBO on prototypes

$$p(y) = \sum_{z_p} p(y \mid z_p)\, p_{\text{proto}}(z_p)$$

$$\geq \quad \sum_{z_p} \log p(y \mid z_p)\, q(z_p) - KL\left(q(z_p) \,\|\, p_{\text{proto}}(z_p)\right)$$

$$q(z_p) \approx p(z_p \mid y) \text{ ?}$$

y = output sentence **z**p = prototype sentence **z**e = edit vector

q(z) over prototypes

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over prototypes

Question

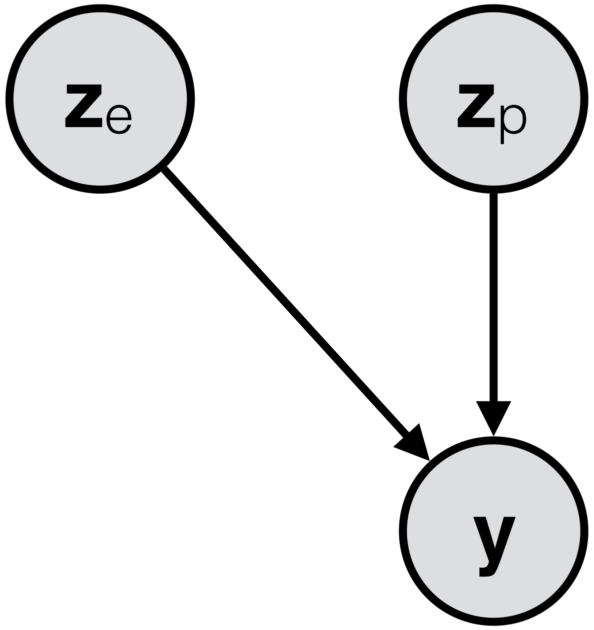
$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

q(z) over prototypes

Question

$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$

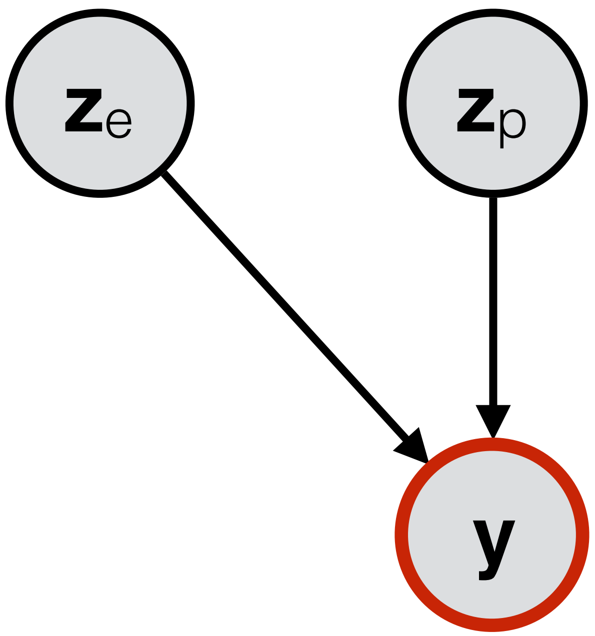


y = output sentence **z**p = prototype sentence **z**e = edit vector

q(z) over prototypes

Question

$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$

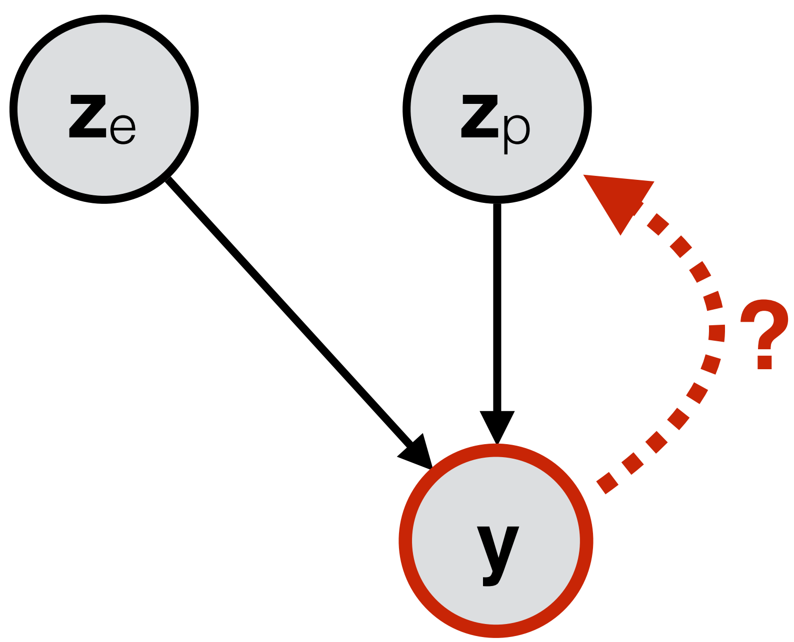


y = output sentence **z**p = prototype sentence **z**e = edit vector

q(z) over prototypes

Question

$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$

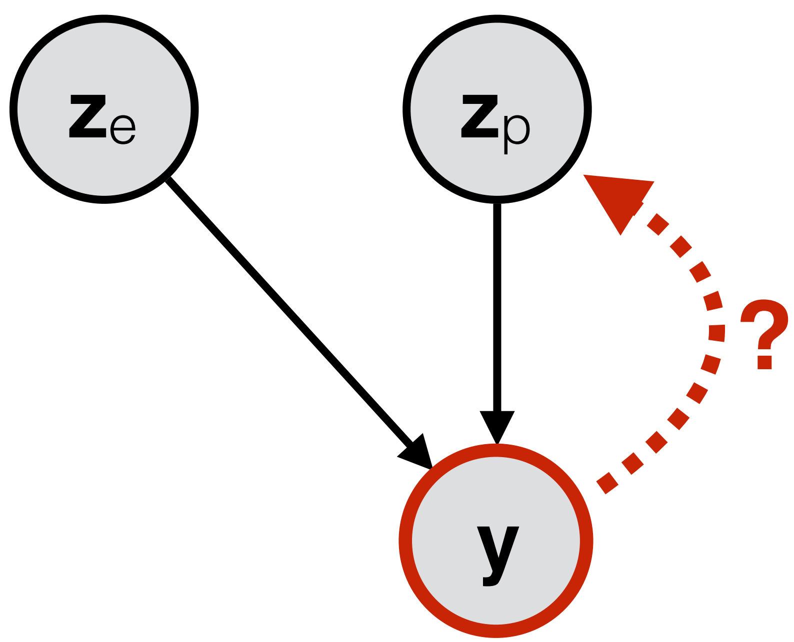


$\mathbf{y}$ = output sentence $\mathbf{z}_{\mathrm{p}}$ = prototype sentence $\mathbf{z}_{\mathrm{e}}$ = edit vector

q(z) over prototypes

Question

$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$



Answer

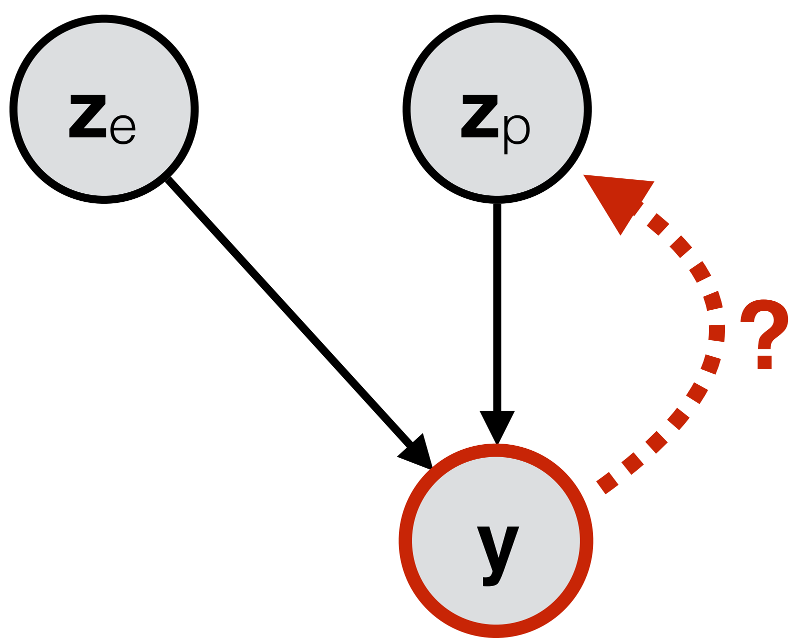
prototype $\mathbf{z}_p$ was probably not too different from $\mathbf{y}$.

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over prototypes

Question

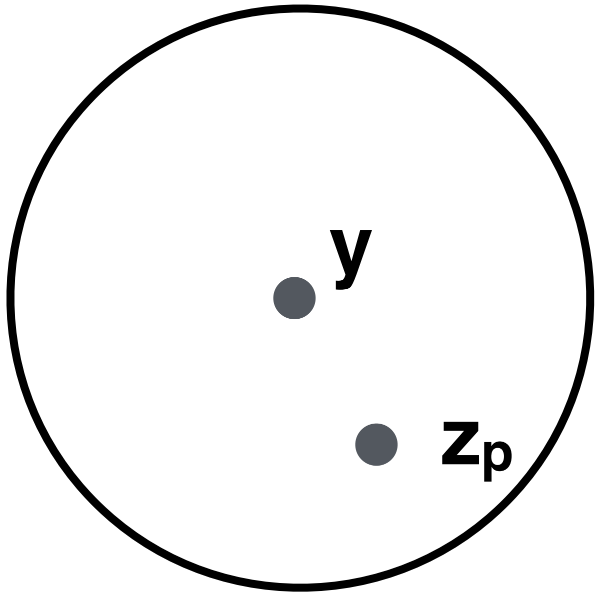
$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$



Answer

prototype **z**p was probably not too different from **y**.

$$\mathcal{N}\left(y\right) =$$

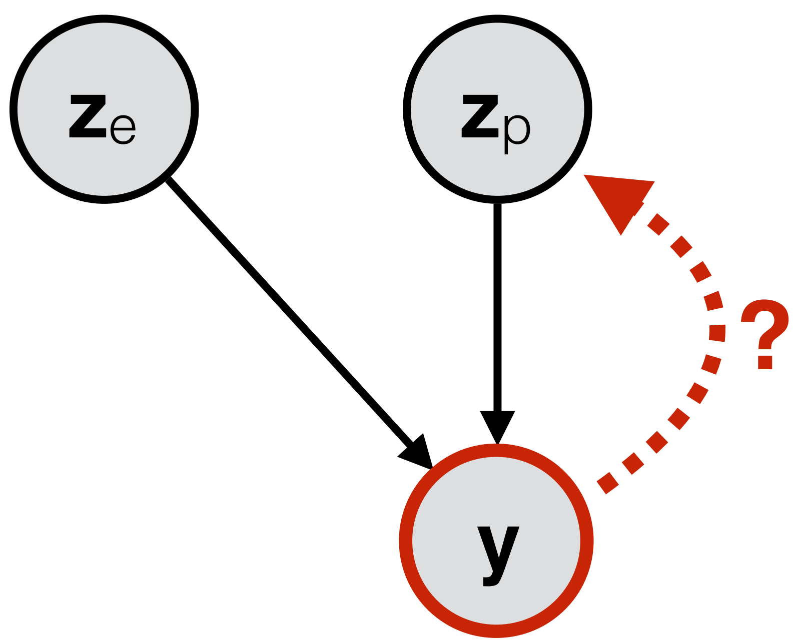


y = output sentence **z**p = prototype sentence **z**e = edit vector

q(z) over prototypes

Question

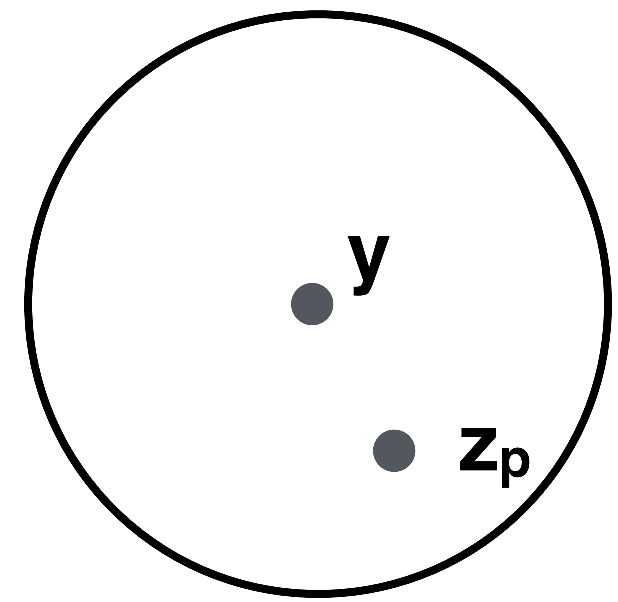
$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$



Answer

prototype $\mathbf{z}_p$ was probably not too different from **y**.

$$\mathcal{N}(y) =$$



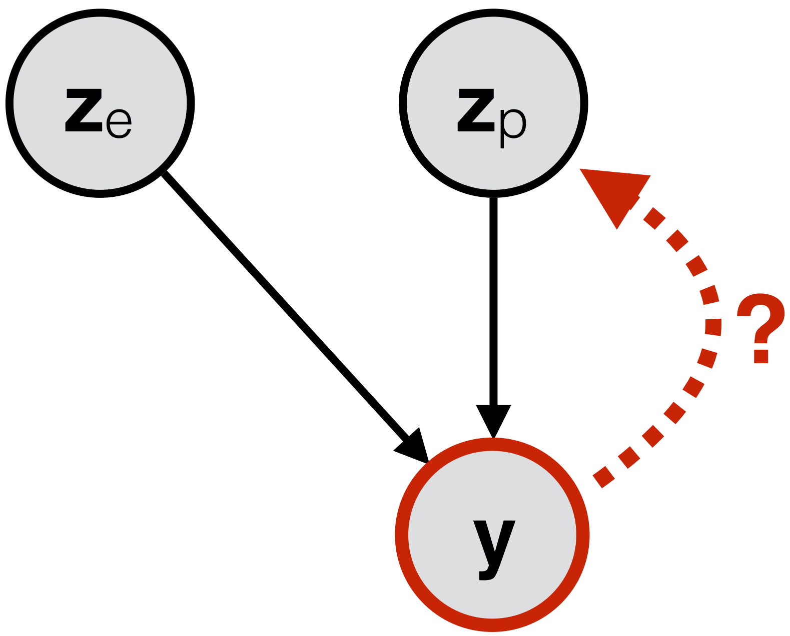
N(y) = all sentences with high token overlap

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over prototypes

Question

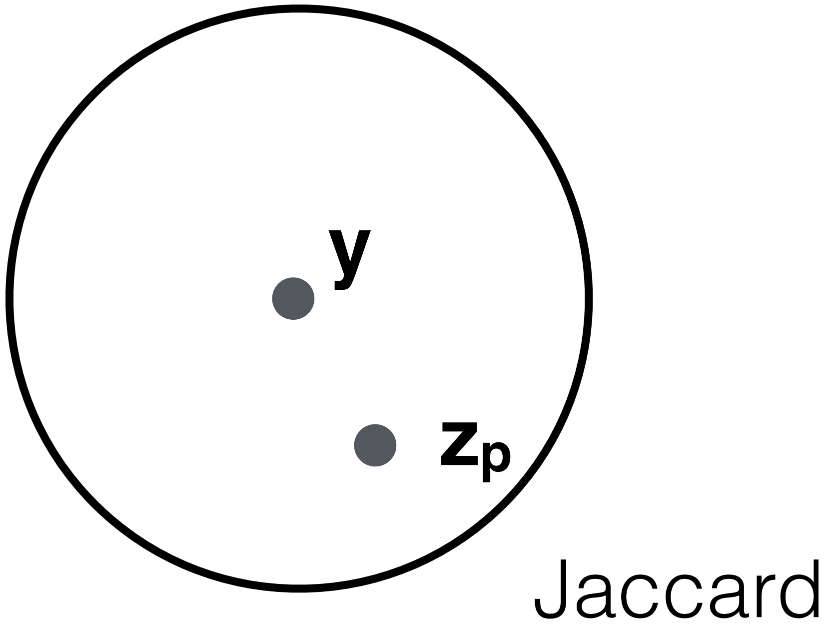
$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$



Answer

prototype $\mathbf{z}_\mathrm{p}$ was probably not too different from **y**.

$$\mathcal{N}\left(y\right) =$$



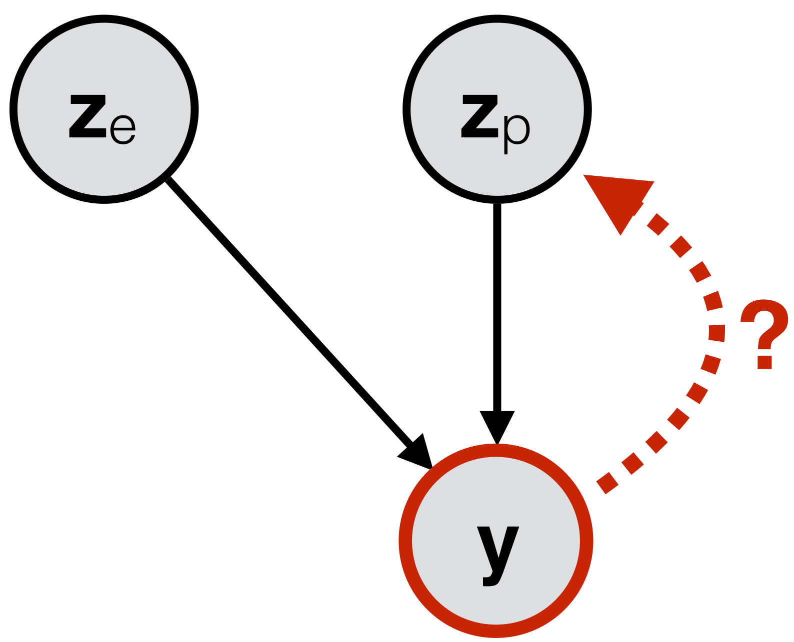
N(y) = all sentences with high token overlap

y = output sentence $\mathbf{z}_\mathrm{p}$ = prototype sentence $\mathbf{z}_\mathrm{e}$ = edit vector

q(z) over prototypes

Question

$$q\left(z_p\right) \approx p\left(z_p \mid y\right)$$

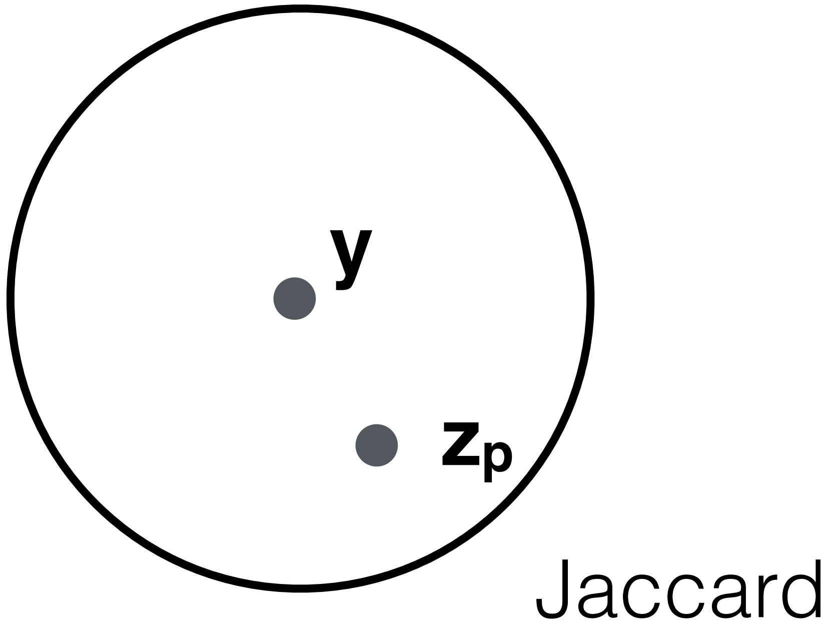


Answer

prototype $\mathbf{z}_p$ was probably not too different from **y**.

$$q(z_p) := \mathrm{Uniform}(\mathcal{N}(y))$$

$$\mathcal{N}\left(y\right) =$$



N(y) = all sentences with high token overlap

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over prototypes

$$\text{ELBO} = \sum_{z_p} \log p\left(y \mid z_p\right) q\left(z_p\right) - KL\left(q\left(z_p\right) \| p_{\text{proto}}\left(z_p\right)\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

q(z) over prototypes

$$\text{ELBO} = \sum_{z_p} \log p\left(y \mid z_p\right) q\left(z_p\right) - KL\left(q\left(z_p\right) \| p_{\text{proto}}\left(z_p\right)\right)$$

$$\shortparallel$$

$$\frac{1}{\left|\mathcal{N}\left(y\right)\right|} \sum_{z_p \in \mathcal{N}(y)} \log p\left(y \mid z_p\right) + C$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

q(z) over prototypes

$$\text{ELBO} = \sum_{z_p} \log p\left(y \mid z_p\right) q\left(z_p\right) - KL\left(q\left(z_p\right) \| p_{\text{proto}}\left(z_p\right)\right)$$

$$=$$

$$\frac{1}{\left|\mathcal{N}\left(y\right)\right|} \sum_{z_p \in \mathcal{N}(y)} \log p\left(y \mid z_p\right) + C$$

Looks like typical **sequence-to-sequence** objective

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over prototypes

$$\text{ELBO} = \sum_{z_p} \log p\left(y \mid z_p\right) q\left(z_p\right) - KL\left(q\left(z_p\right) \| p_{\text{proto}}\left(z_p\right)\right)$$

$$=$$

$$\frac{1}{\left|\mathcal{N}\left(y\right)\right|} \sum_{z_p \in \mathcal{N}\left(y\right)} \log p\left(y \mid z_p\right) + C$$

Looks like typical **sequence-to-sequence** objective

prototype $\mathbf{z}_p$ —> output $\mathbf{y}$

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over prototypes

$$\text{ELBO} = \sum_{z_p} \log p(y \mid z_p)\, q(z_p) - KL\left(q(z_p) \,\|\, p_{\text{proto}}(z_p)\right)$$

$$\|$$

$$\frac{1}{|\mathcal{N}(y)|} \sum_{z_p \in \mathcal{N}(y)} \log p(y \mid z_p) + C$$

Looks like typical **sequence-to-sequence** objective

prototype $\mathbf{z}_p$ —> output $\mathbf{y}$

✓ **bias towards small edits**

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

q(z) over prototypes

$$\text{ELBO} = \sum_{z_p} \log p\left(y \mid z_p\right) q\left(z_p\right) - KL\left(q\left(z_p\right) \| p_{\text{proto}}\left(z_p\right)\right)$$

$$\|$$

$$\frac{1}{|\mathcal{N}(y)|} \sum_{z_p \in \mathcal{N}(y)} \log p\left(y \mid z_p\right) + C$$

Looks like typical **sequence-to-sequence** objective

prototype $\mathbf{z}_p$ —> output $\mathbf{y}$

✔ **bias towards small edits**

✔ **computationally tractable**

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Training objective

maximize

$$p\left(y\right) = \sum_{z_p} p\left(y \mid z_p\right) p_{\text{proto}}\left(z_p\right)$$

expensive

$$\int_{z_e} p_{\text{editor}}\left(y \mid z_p, z_e\right) p_{\text{edit}}\left(z_e\right) dz_e$$

intractable

y = output sentence **z**p = prototype sentence **z**e = edit vector

ELBO on edit vectors

$$\log p\left(y \mid z_p\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO on edit vectors

$$\log p\left(y \mid z_p\right)$$

$$\geq \quad E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right] - KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO on edit vectors

$$\log p\left(y \mid z_p\right)$$

$$\geq \quad \underbrace{E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right]}_{\text{reconstruction\_cost}} - KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)$$

y = output sentence **z**p = prototype sentence **z**e = edit vector

ELBO on edit vectors

$$\log p\left(y \mid z_p\right)$$

$$\geq \quad \underbrace{E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right]}_{\text{reconstruction\_cost}} - \underbrace{KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)}_{\text{KL\_penalty}}$$

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

ELBO on edit vectors

sample **z**$_e$ from **q**(**z**$_e$)

$$\log p\left(y \mid z_p\right)$$

$$\geq \quad \underbrace{E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right]}_{\text{reconstruction\_cost}} - \underbrace{KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)}_{\text{KL\_penalty}}$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO on edit vectors

sample $\mathbf{z}_e$ from $\mathbf{q}(\mathbf{z}_e)$

$$\log p\left(y \mid z_p\right)$$

$\mathbf{z}_p$, $\mathbf{z}_e$ —> $\mathbf{y}$

$$\geq \quad \underbrace{E_{z_e\sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right]}_{\text{reconstruction\_cost}} - \underbrace{KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)}_{\text{KL\_penalty}}$$

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

ELBO on edit vectors

sample **z_e** from **q**(**z_e**)

$$\log p\left(y \mid z_p\right)$$

z_p, **z_e** —> **y**

$$\geq \quad \underbrace{E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right]}_{\text{reconstruction\_cost}} - \underbrace{KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)}_{\text{KL\_penalty}}$$

measures how well we can reconstruct **y** from prototype **z_p** and edit **z_e**

y = output sentence **z_p** = prototype sentence **z_e** = edit vector

ELBO on edit vectors

sample $\mathbf{z}_e$ from $\mathbf{q}(\mathbf{z}_e)$

$$\log p\left(y \mid z_p\right)$$

$\mathbf{z}_p$, $\mathbf{z}_e$ —> $\mathbf{y}$

$$\geq \quad \underbrace{E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right]}_{\text{reconstruction\_cost}} - \underbrace{KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)}_{\text{KL\_penalty}}$$

measures how well we can reconstruct **y** from prototype $\mathbf{z}_p$ and edit $\mathbf{z}_e$

measures difference between **q** and edit prior $\mathbf{p}_{\text{edit}}$

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

ELBO on edit vectors

$$\log p\left(y \mid z_p\right)$$

$$\geq \quad E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right] - KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

ELBO on edit vectors

$$\log p\left(y \mid z_p\right)$$

$$\geq \quad E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right] - KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)$$

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$

y = output sentence **z**p = prototype sentence **z**e = edit vector

ELBO on edit vectors

$$\log p\left(y \mid z_p\right)$$

$$\geq \quad E_{z_e \sim q(z_e)}\left[\log p_{\text{editor}}\left(y \mid z_p, z_e\right)\right] - KL\left(q\left(z_e\right) \| p_{\text{edit}}\left(z_e\right)\right)$$

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$ **?**

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

q(z) over edits

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over edits

Question

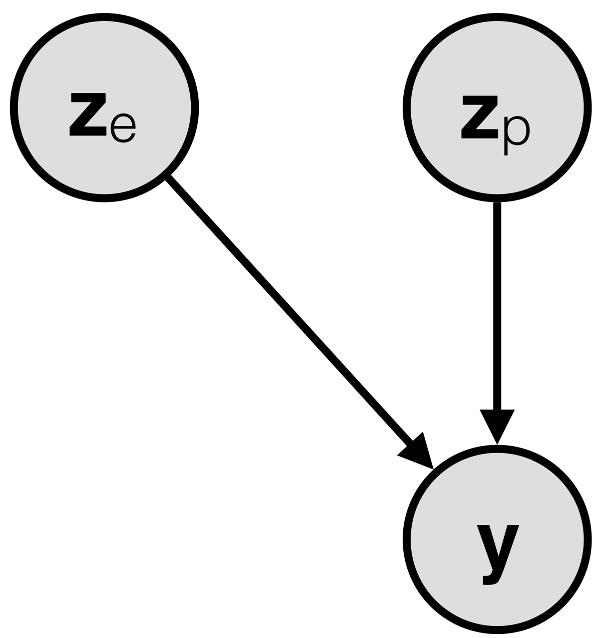
$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

q(z) over edits

Question

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$

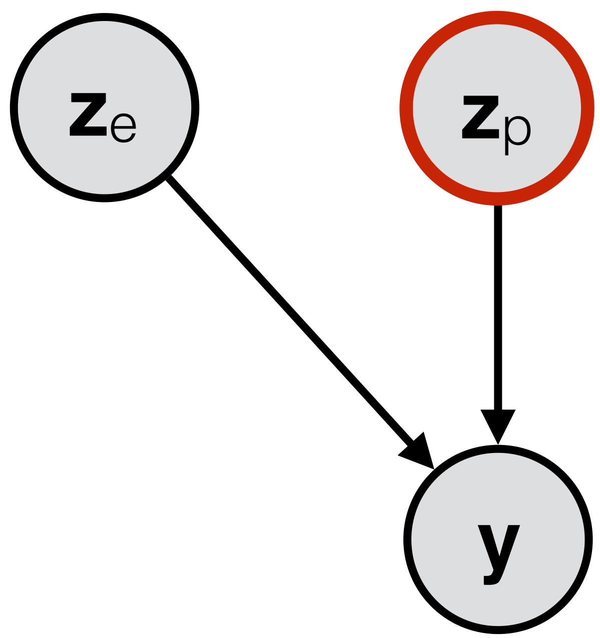


y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

q(z) over edits

Question

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$

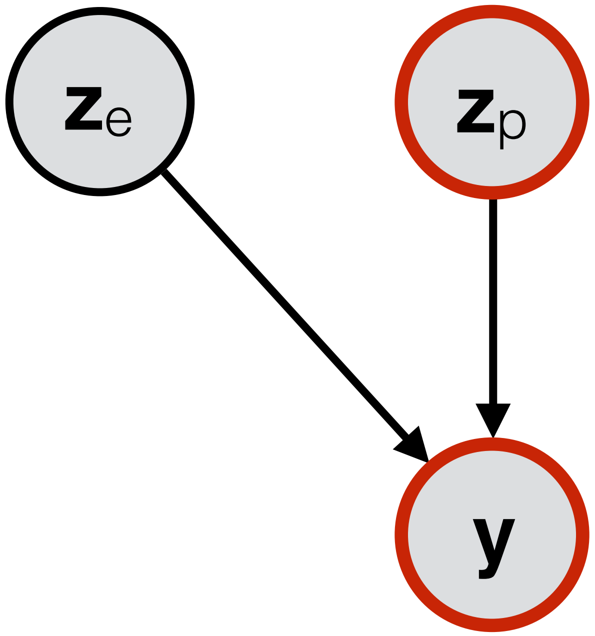


y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over edits

Question

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$

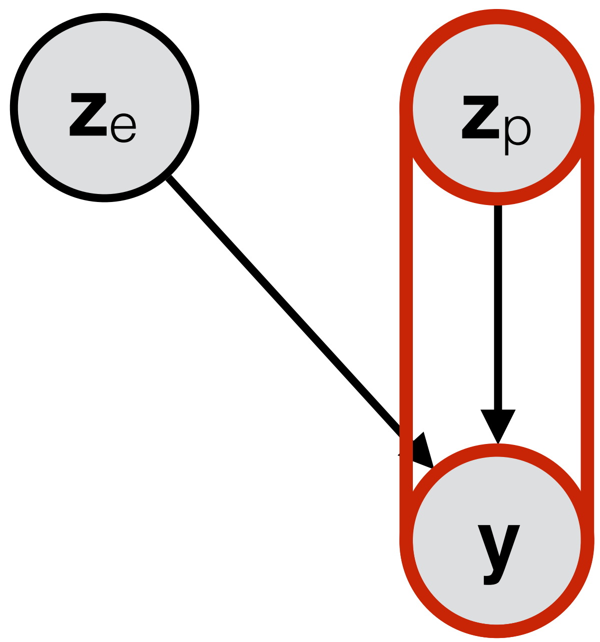


y = output sentence **z**p = prototype sentence **z**e = edit vector

q(z) over edits

Question

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$

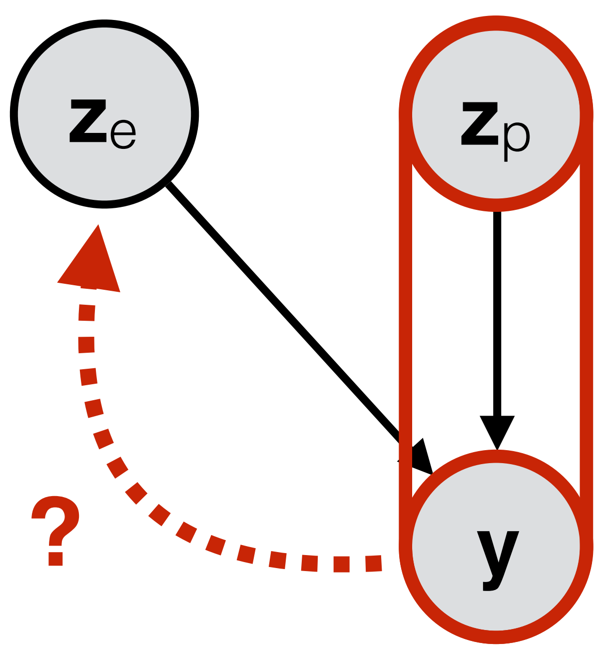


$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over edits

Question

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$



y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

q(z) over edits

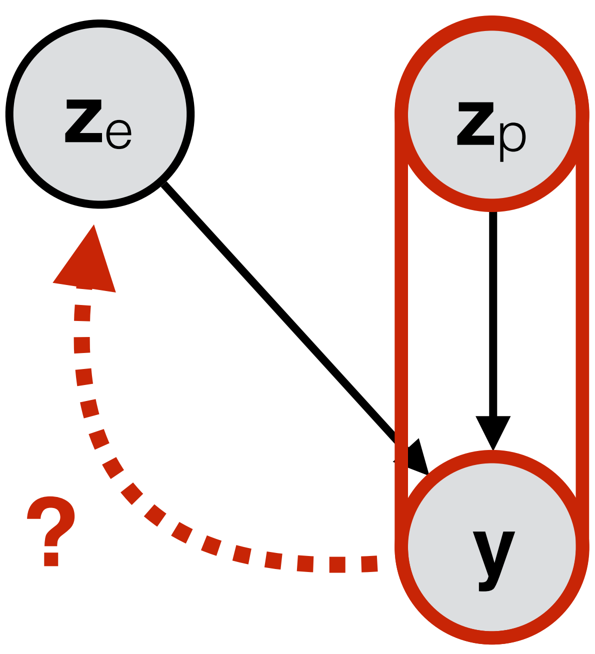
Question

$$q\left(z_e\right) \approx p\left(z_e \mid y, z_p\right)$$

Answer

Compare the two sentences.

Figure out which words were **inserted** and **deleted**.
Then sum their word vectors.



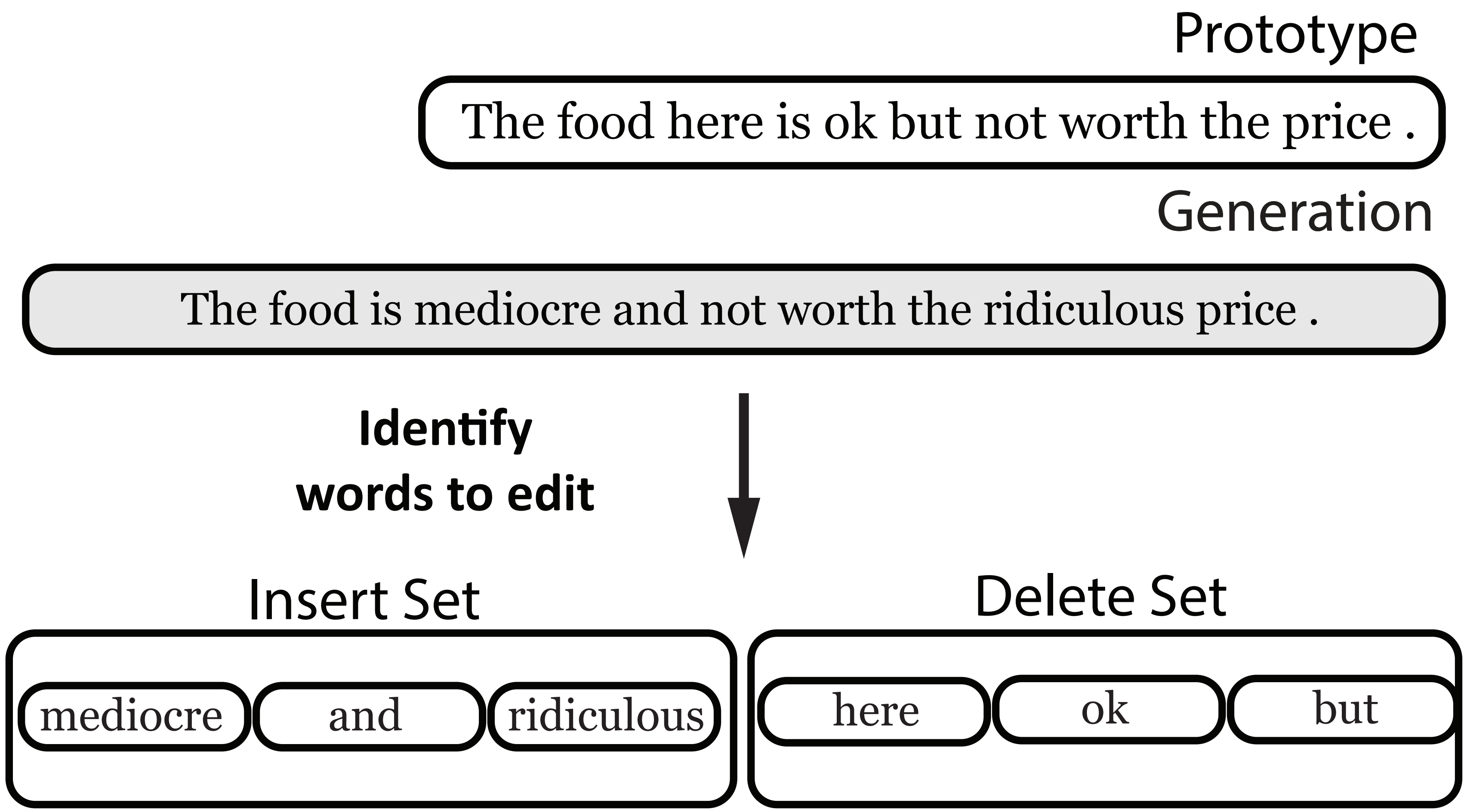
y = output sentence **z_p** = prototype sentence **z_e** = edit vector

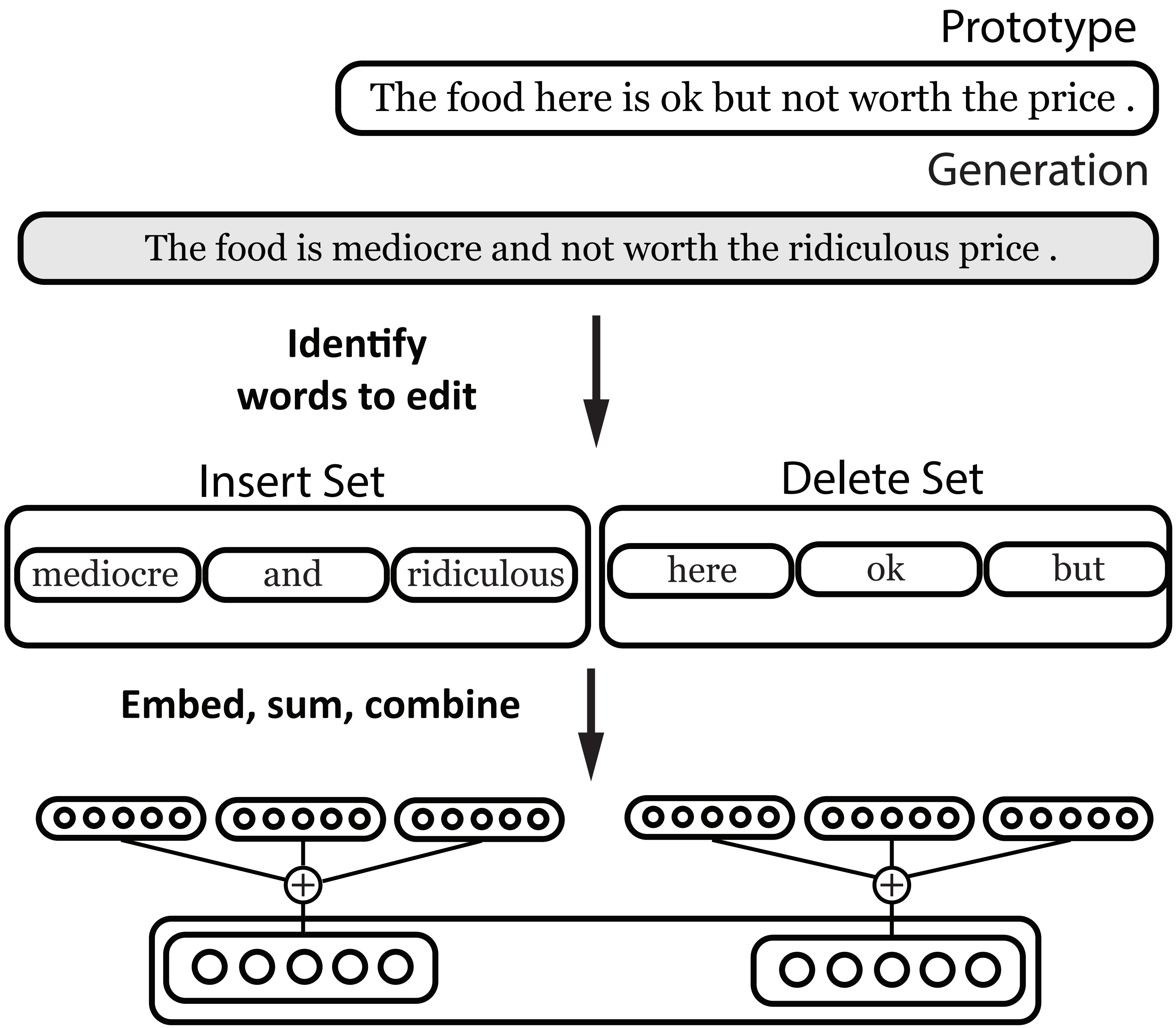
Prototype

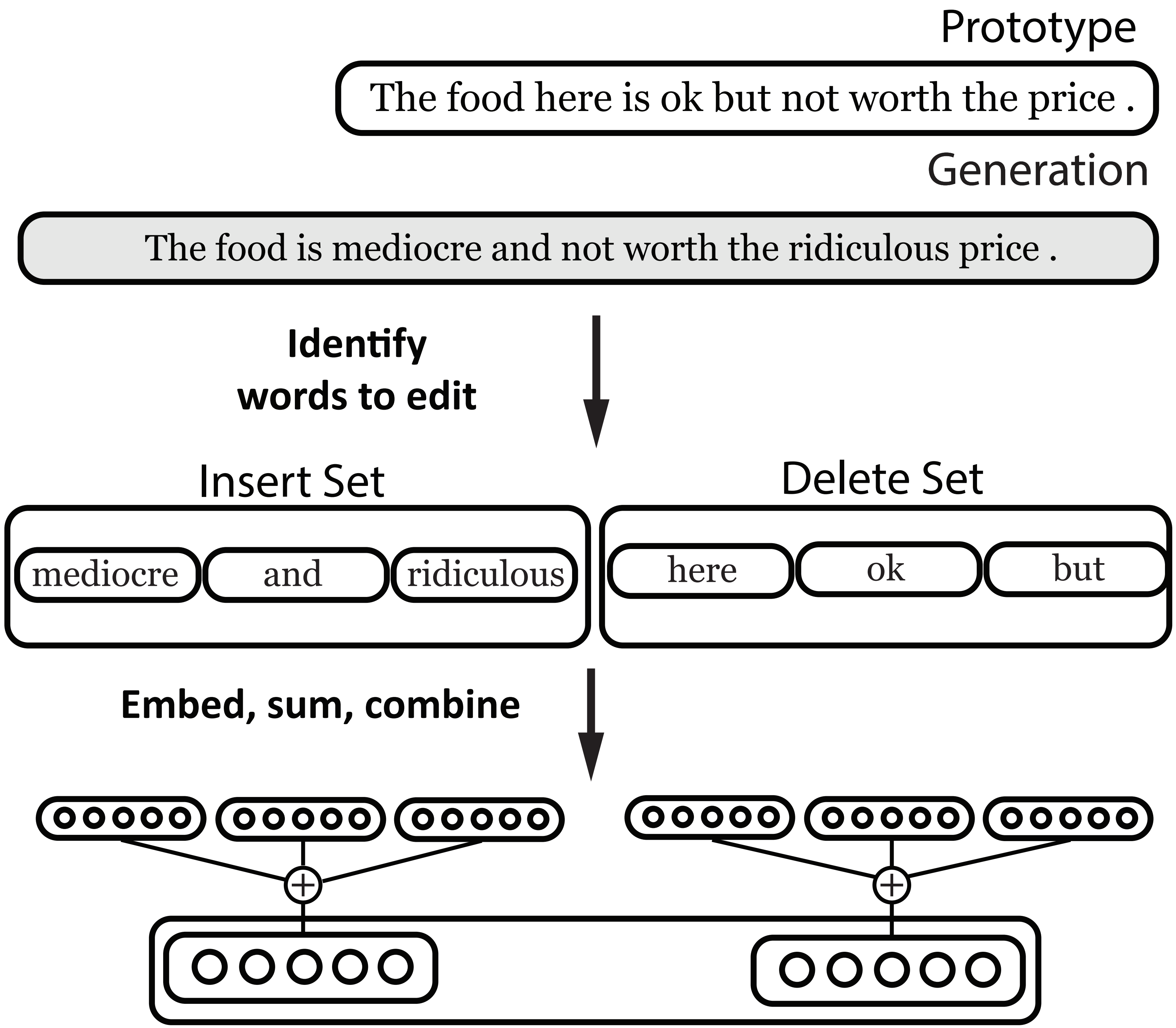
The food here is ok but not worth the price .

Generation

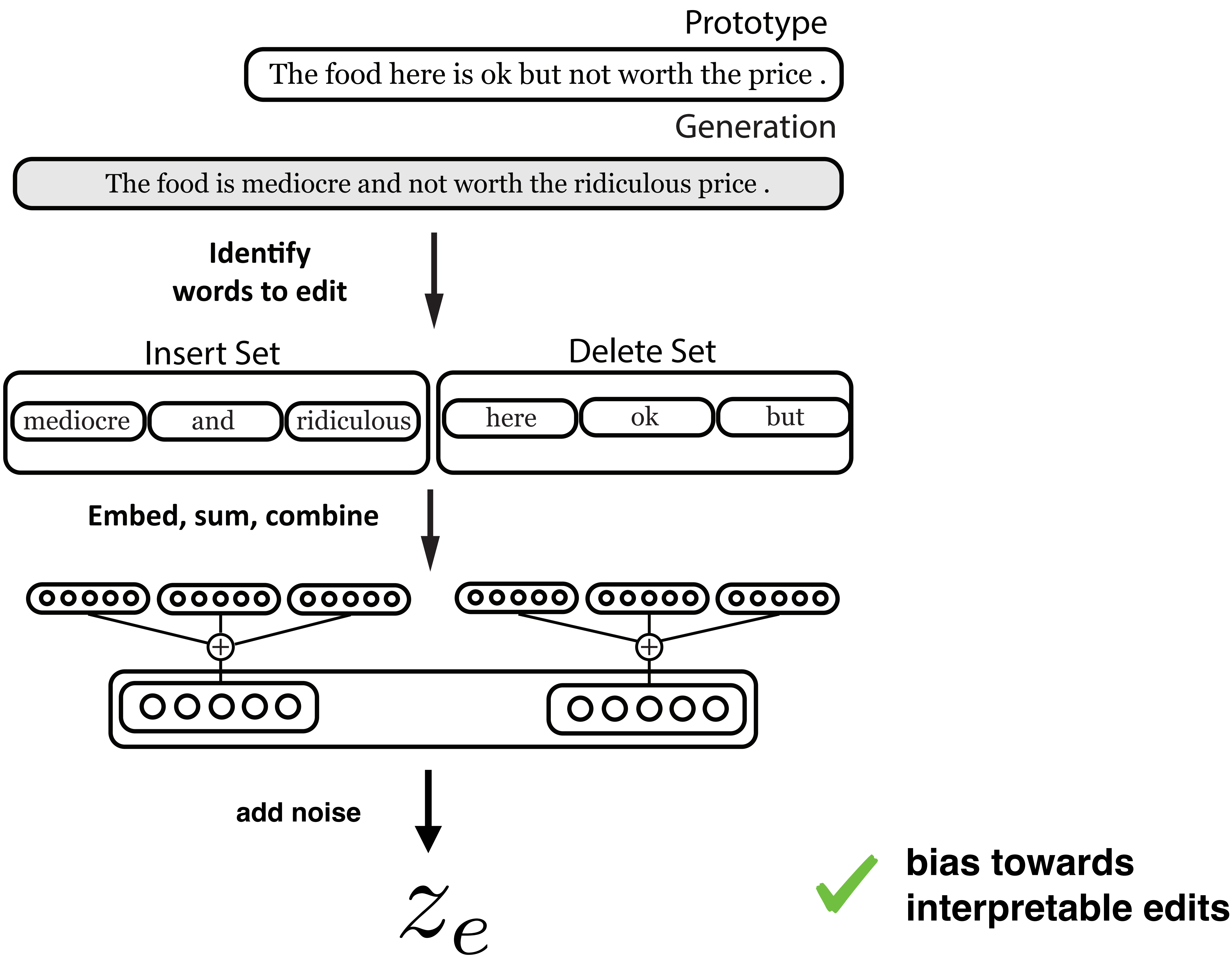
The food is mediocre and not worth the ridiculous price .


Prototype
The food here is ok but not worth the price .
Generation
The food is mediocre and not worth the ridiculous price .
Identify words to edit
Insert Set
mediocre
and
ridiculous
Delete Set
here
ok
but

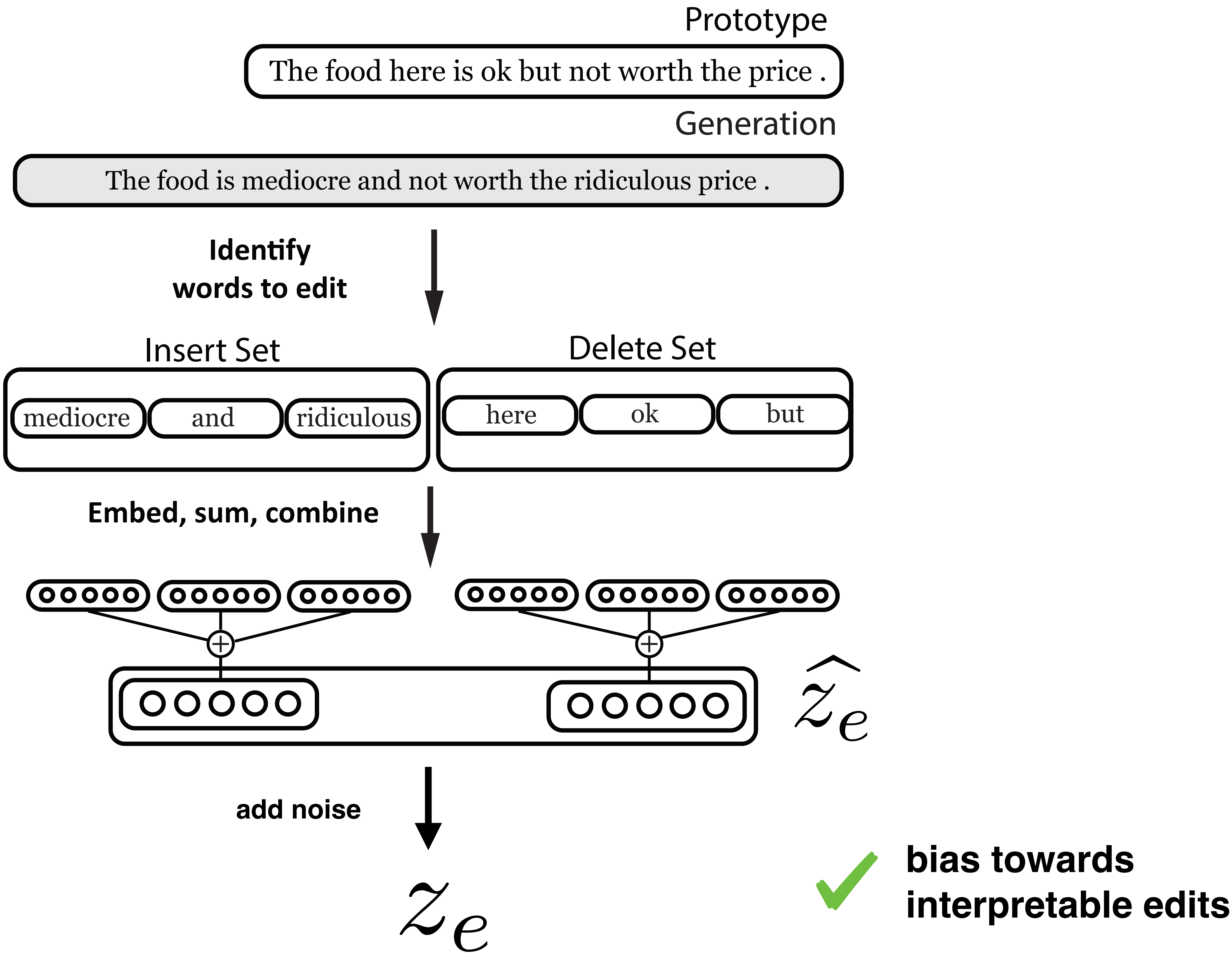

Prototype
The food here is ok but not worth the price .
Generation
The food is mediocre and not worth the ridiculous price .
Identify words to edit
Insert Set
mediocre
and
ridiculous
Delete Set
here
ok
but
Embed, sum, combine



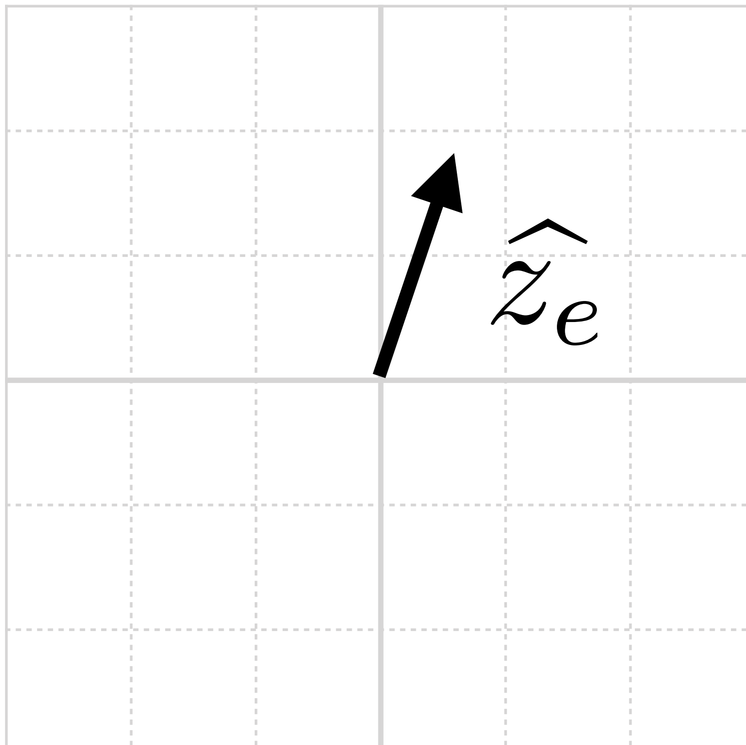
✓ **bias towards interpretable edits**



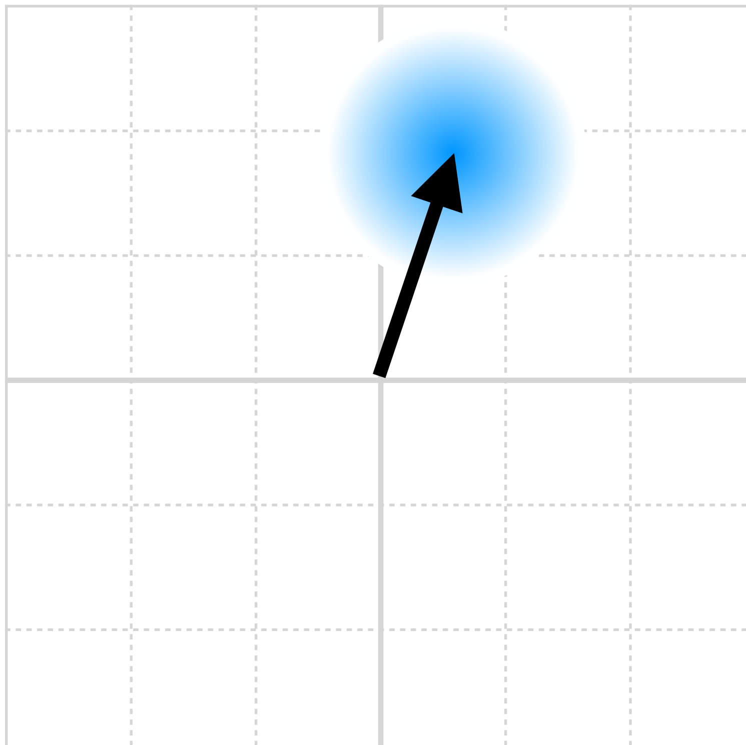
Prototype
The food here is ok but not worth the price .
Generation
The food is mediocre and not worth the ridiculous price .
Identify
words to edit
Insert Set
mediocre
and
ridiculous
Delete Set
here
ok
but
Embed, sum, combine
add noise
z_e
bias towards
interpretable edits


Prototype
The food here is ok but not worth the price .
Generation
The food is mediocre and not worth the ridiculous price .
Identify
words to edit
Insert Set
mediocre
and
ridiculous
Delete Set
here
ok
but
Embed, sum, combine
$\hat{z}_e$
add noise
z_e
bias towards
interpretable edits

How to add noise to $\widehat{z_e}$?

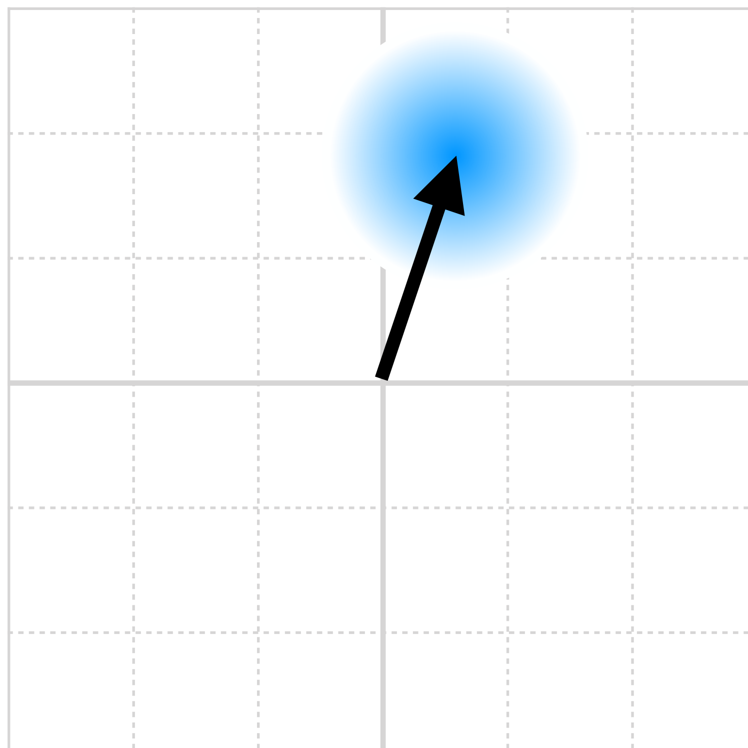


Standard choice (VAE): Gaussian

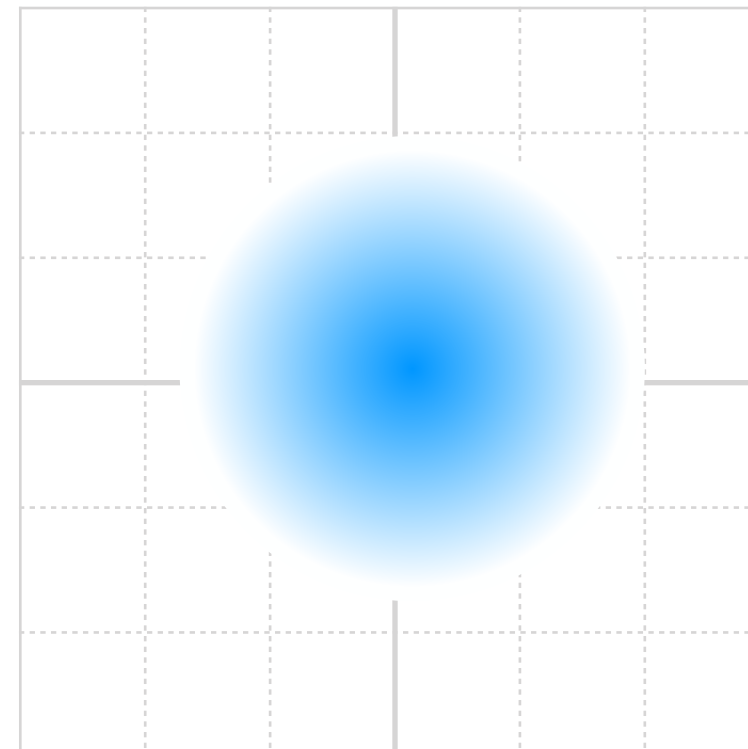


$q\left(z_e\right)$

Standard choice (VAE): Gaussian

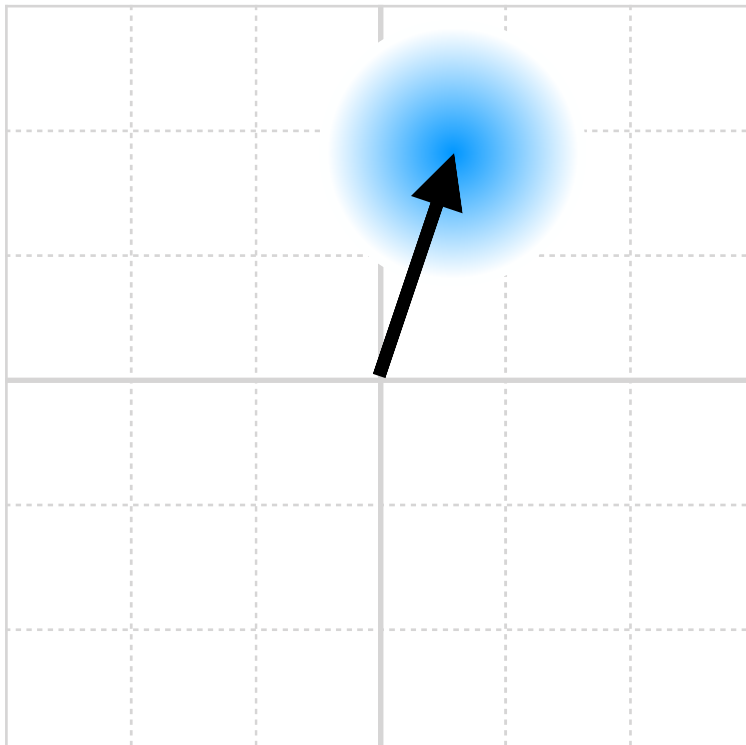


$q\left(z_e\right)$

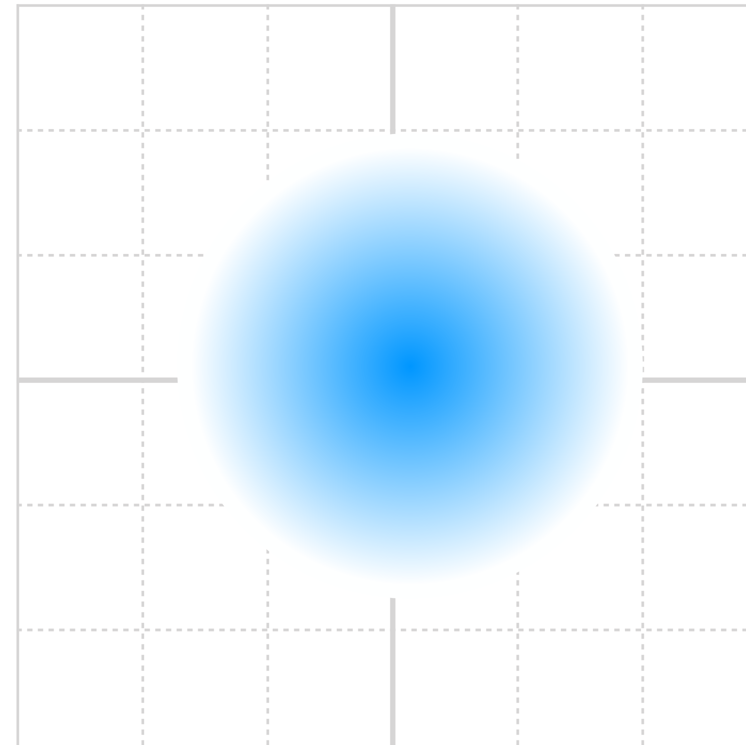


$p_{\text{edit}}\left(z_e\right)$

Standard choice (VAE): Gaussian



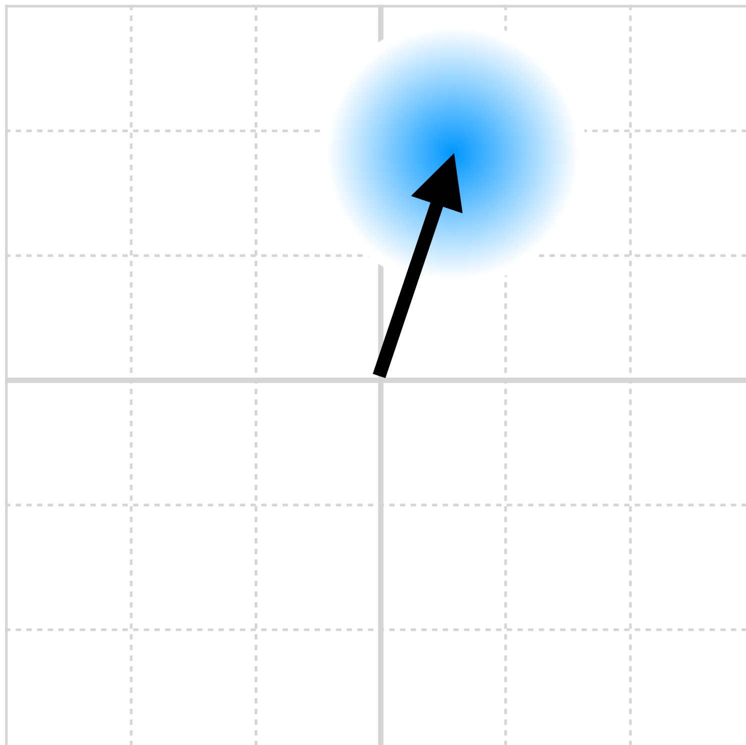
$q(z_e)$



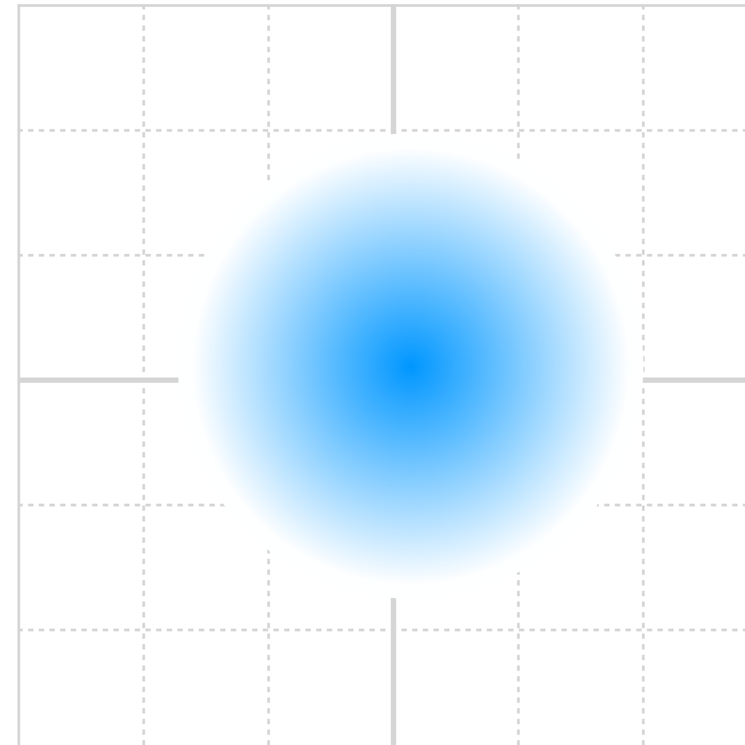
$p_{\text{edit}}(z_e)$

✓ **computationally tractable**

Standard choice (VAE): Gaussian



$q(z_e)$

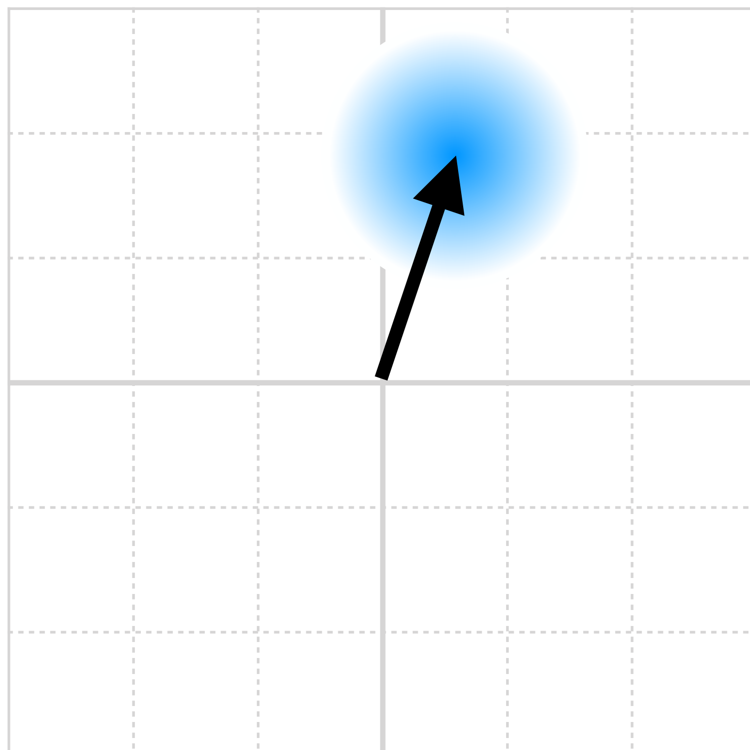


$p_{\text{edit}}(z_e)$

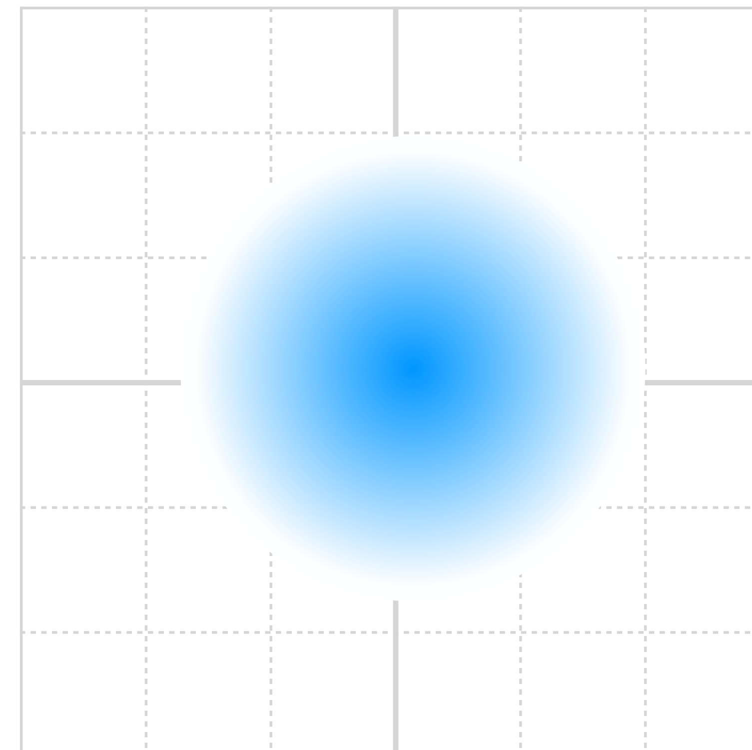
ELBO = reconstruction_cost - KL_penalty

✓ **computationally tractable**

Standard choice (VAE): Gaussian



$q\left(z_e\right)$



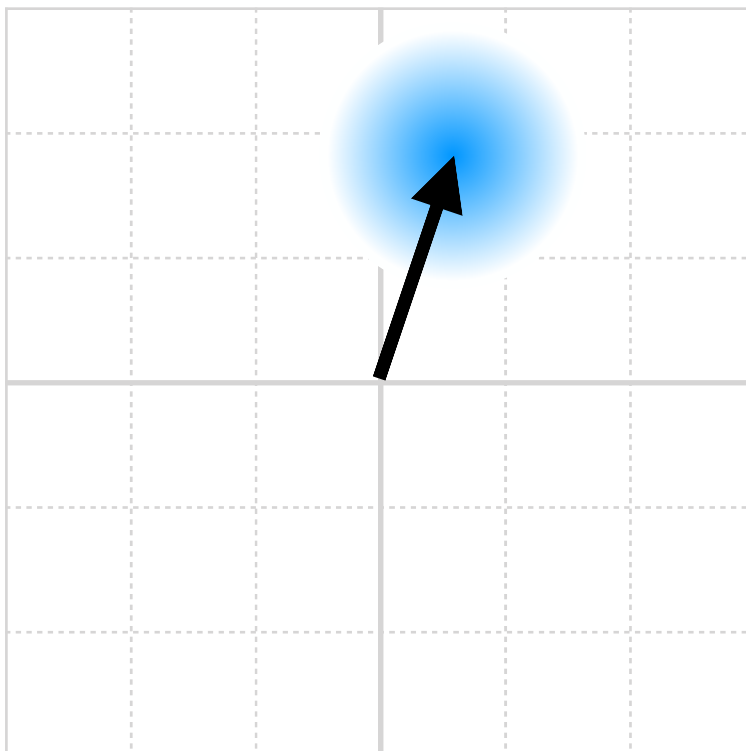
$p_{\mathrm{edit}}\left(z_e\right)$

ELBO = reconstruction_cost - KL_penalty

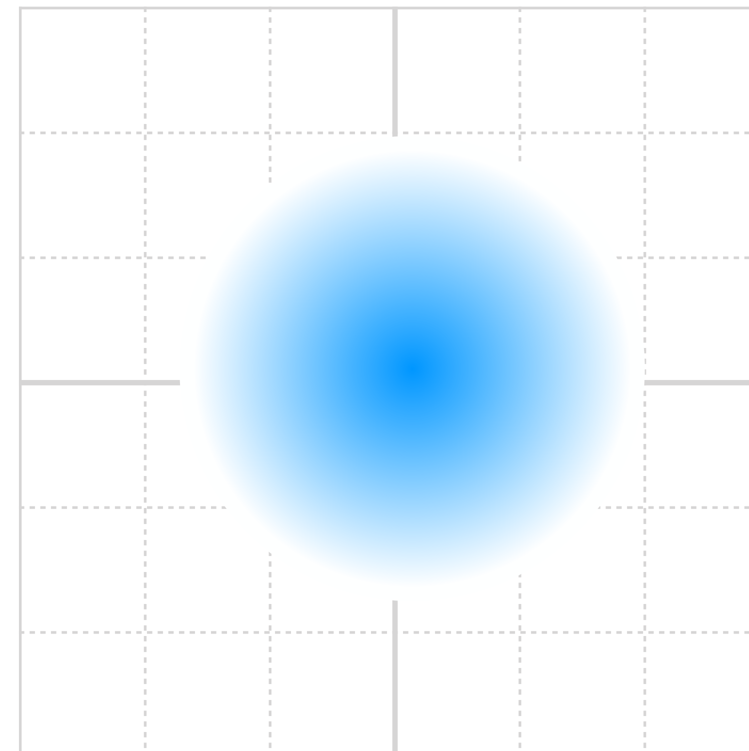
reparameterization trick (VAEs)

✓ **computationally tractable**

Standard choice (VAE): Gaussian



$q\left(z_e\right)$



$p_{\text{edit}}\left(z_e\right)$

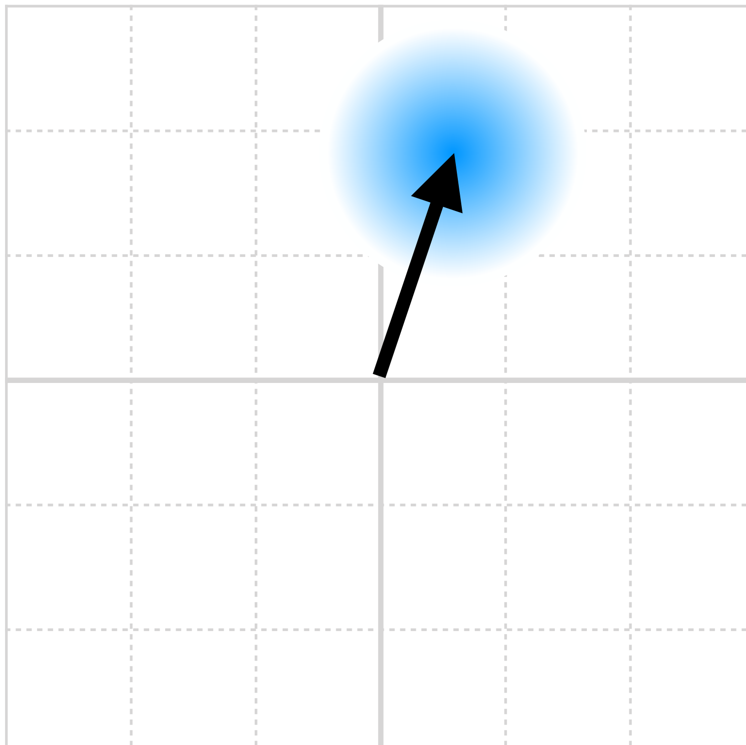
ELBO = reconstruction_cost - KL_penalty

reparameterization trick (VAEs)

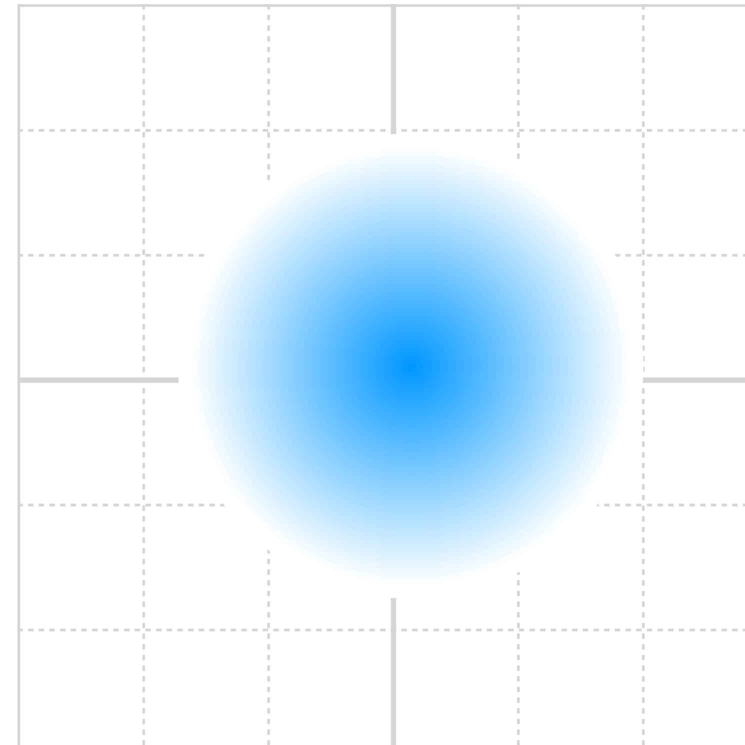
(low-variance MC estimate of gradient)

✓ **computationally tractable**

Standard choice (VAE): Gaussian



$q\left(z_e\right)$



$p_{\text{edit}}\left(z_e\right)$

ELBO = reconstruction_cost - KL_penalty

reparameterization trick (VAEs)

(low-variance MC estimate of gradient)

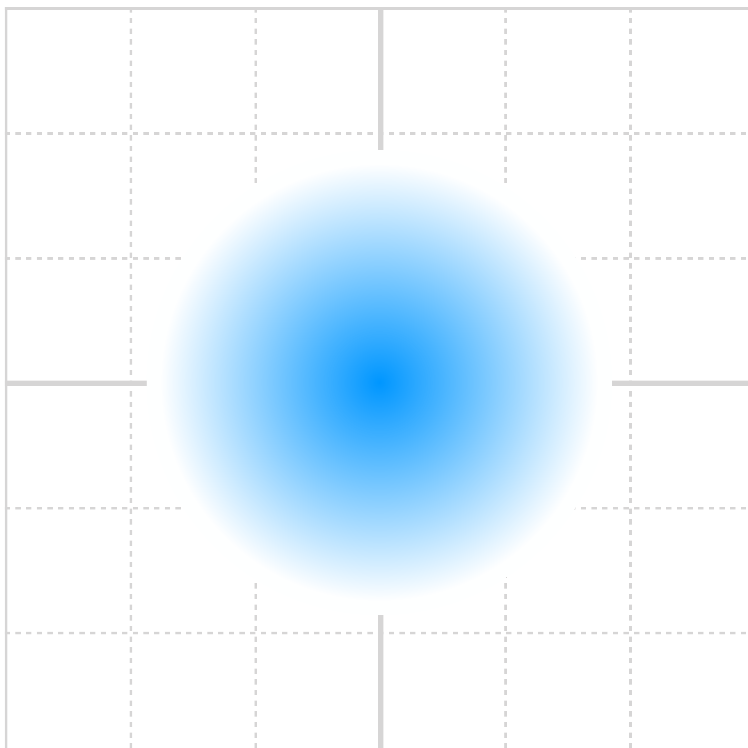
closed form

✓ **computationally tractable**

The problem with a Gaussian prior

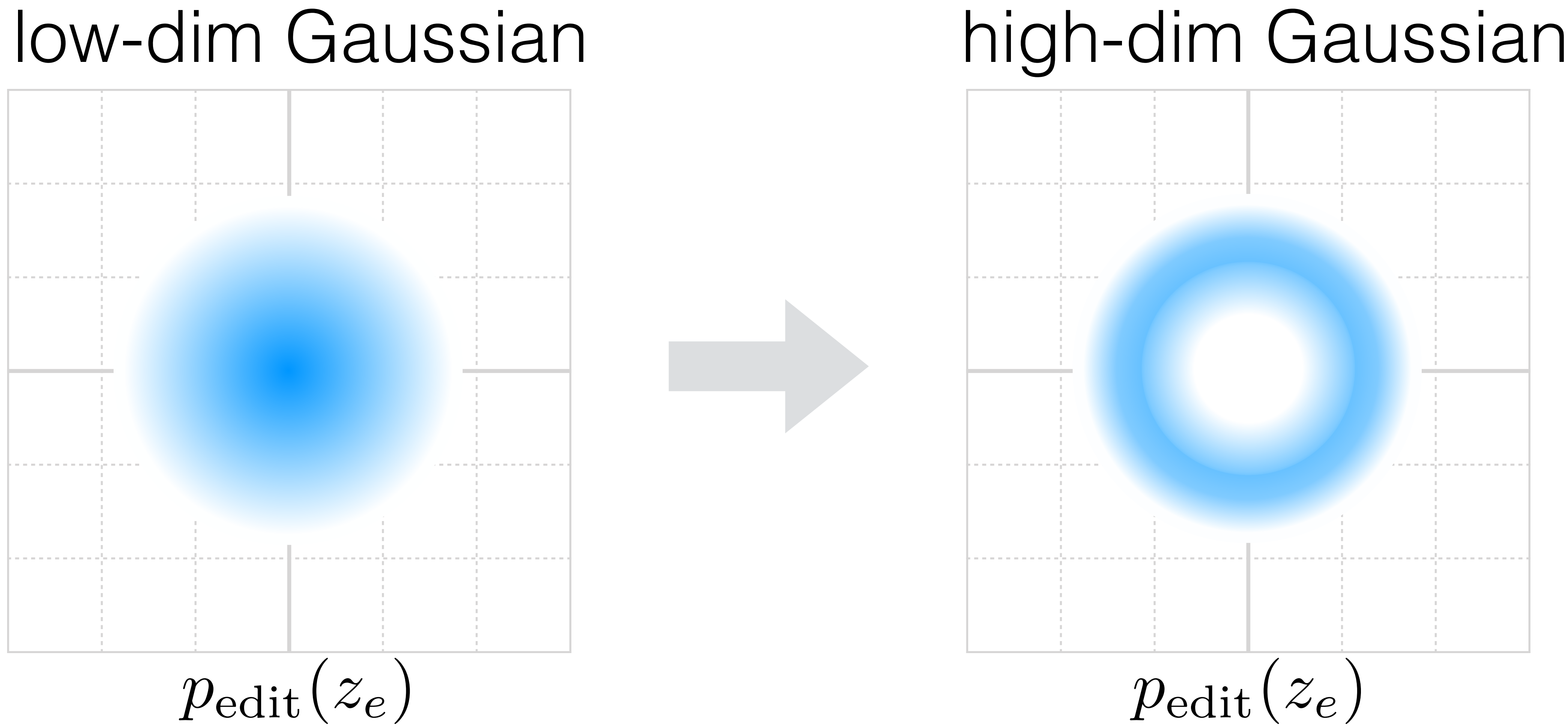
The problem with a Gaussian prior

low-dim Gaussian



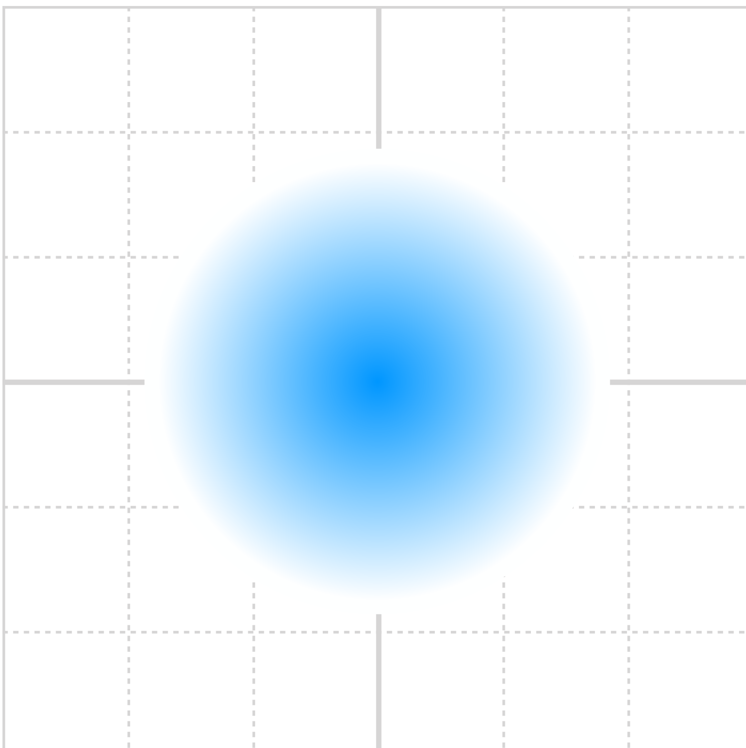
$p_{\text{edit}}(z_e)$

The problem with a Gaussian prior



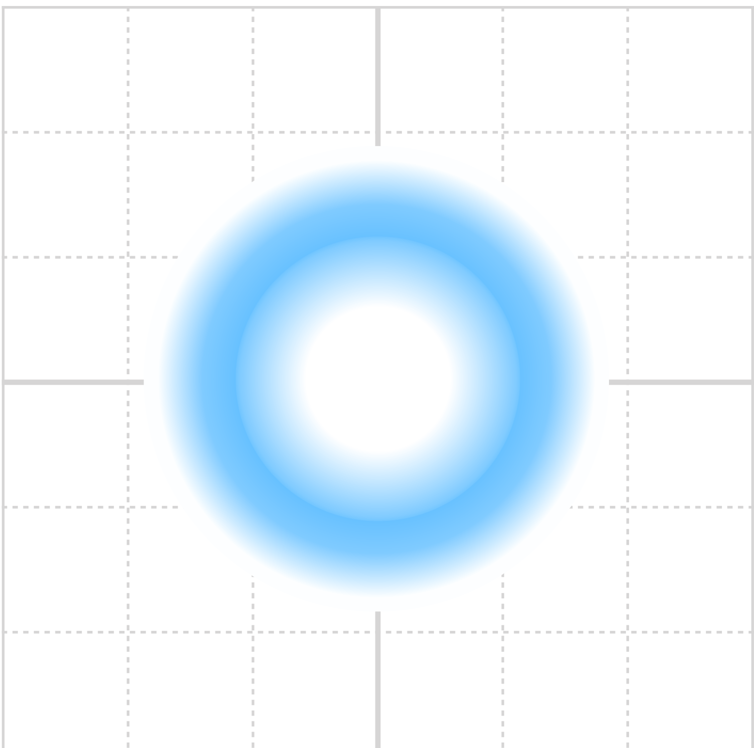
The problem with a Gaussian prior

low-dim Gaussian

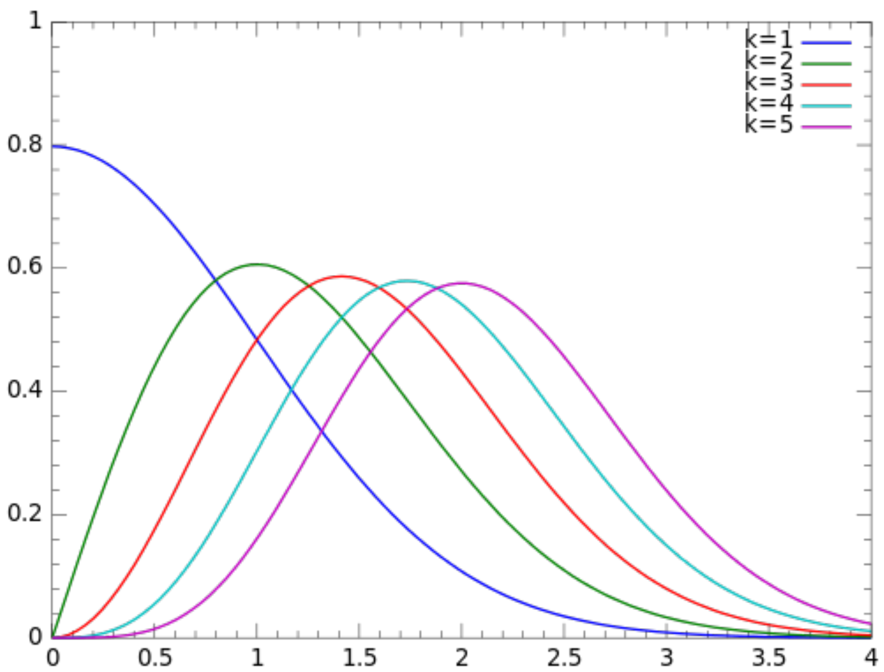


$p_{\text{edit}}(z_e)$

high-dim Gaussian

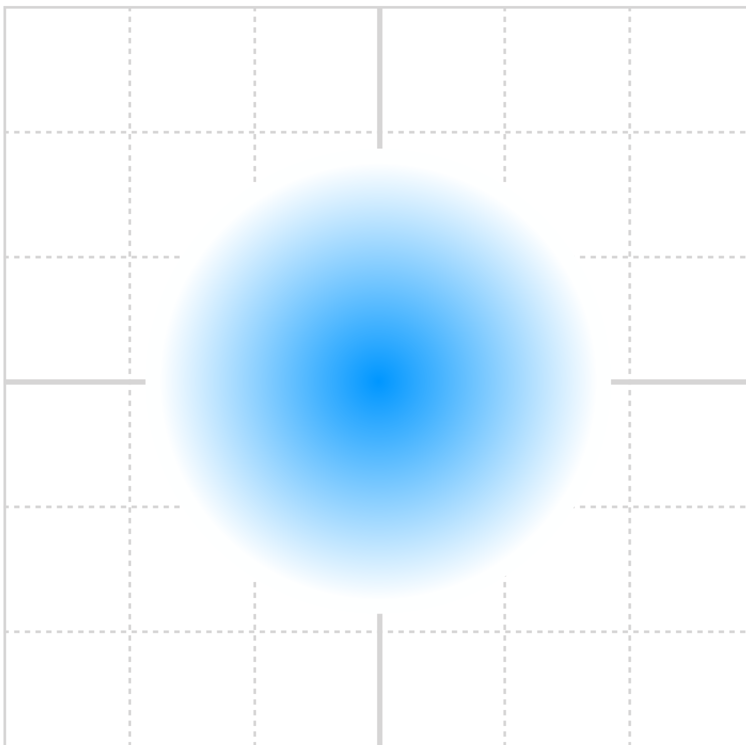


$p_{\text{edit}}(z_e)$



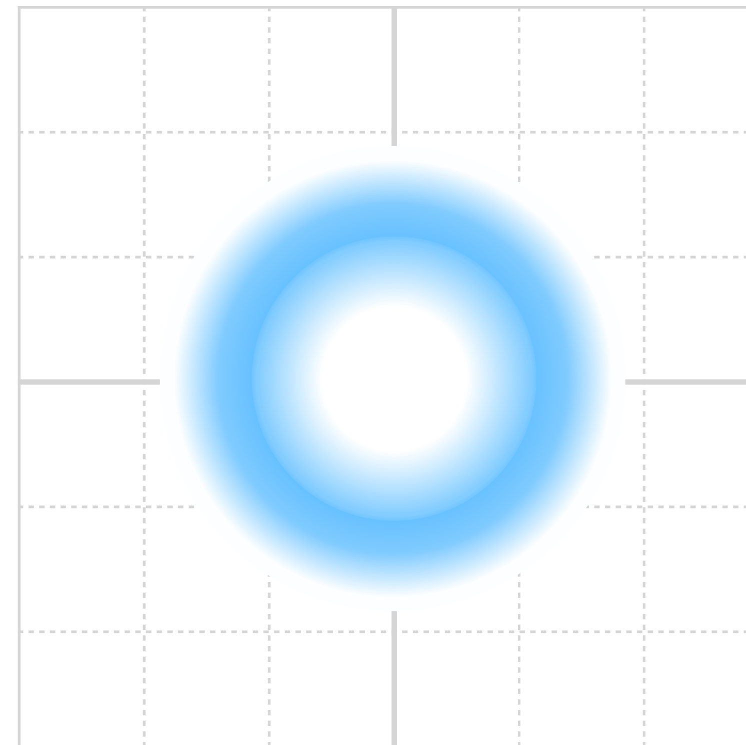
The problem with a Gaussian prior

low-dim Gaussian

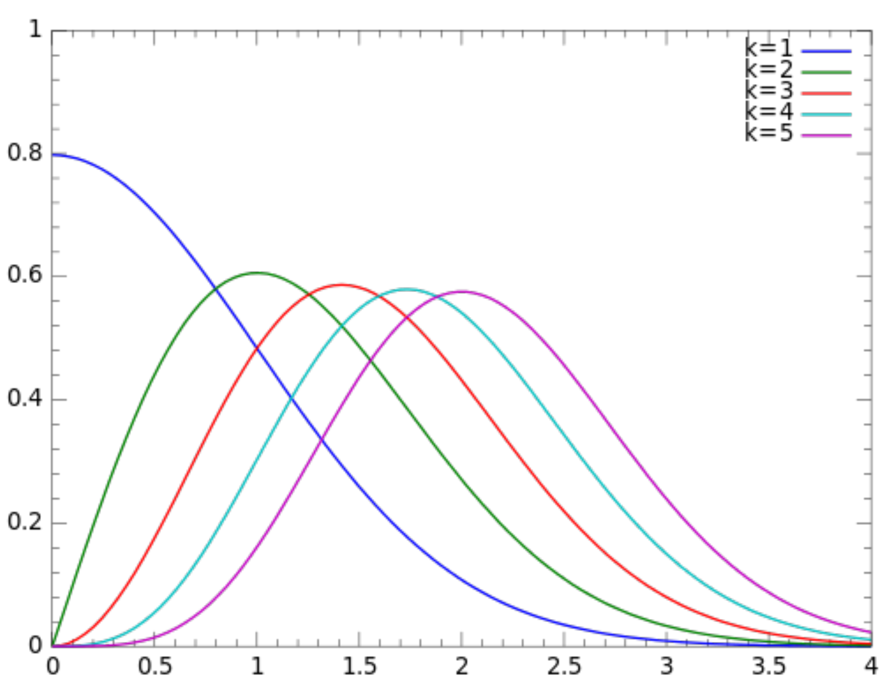


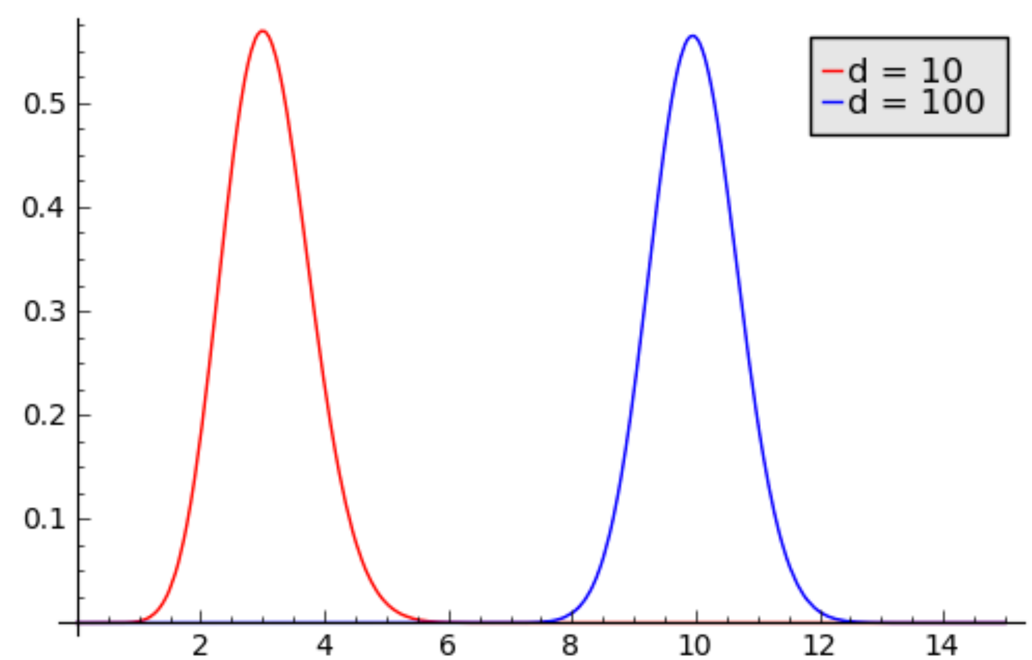
$p_{\text{edit}}(z_e)$

high-dim Gaussian



$p_{\text{edit}}(z_e)$





Better edit prior

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Better edit prior

$$\text{mag} \sim \text{Unif}\,[0, 10]$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Better edit prior

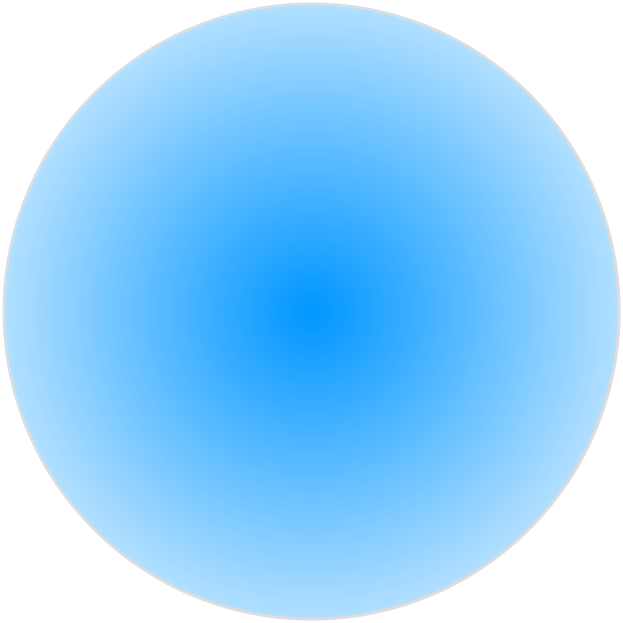
$$\text{mag} \sim \text{Unif}\,[0, 10]$$
$$\text{dir} \sim \text{unif. over sphere}$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Better edit prior

$$\text{mag} \sim \text{Unif}\,[0, 10]$$
$$\text{dir} \sim \text{unif. over sphere}$$

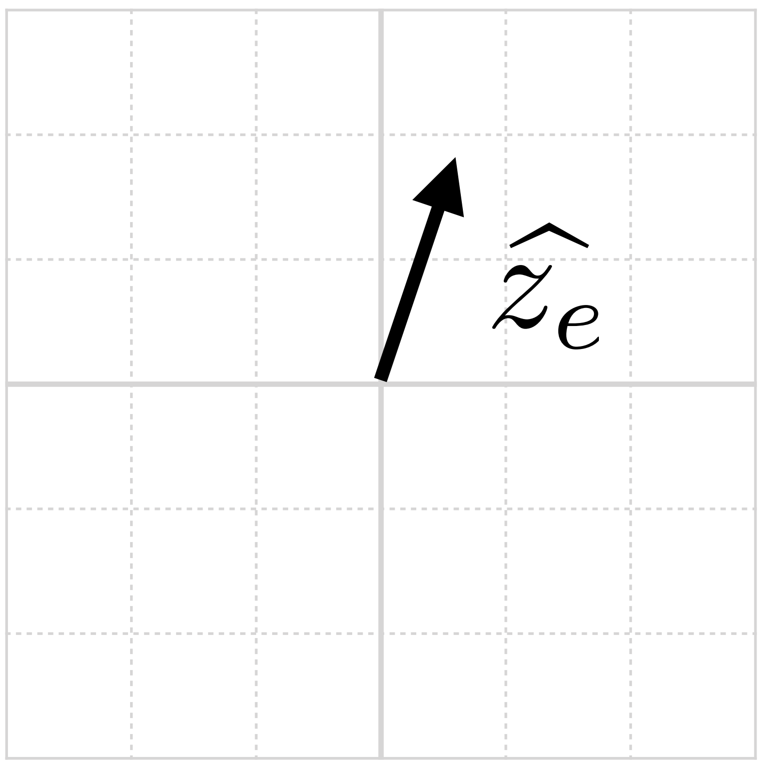
$p_{\text{edit}}\left(z_e\right)$



y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

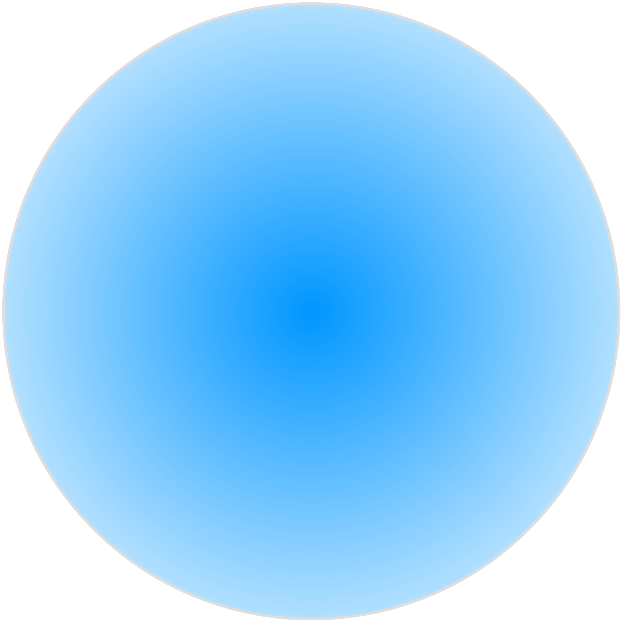
Better edit prior

$q\left(z_e\right)$**?**



$$\mathrm{mag} \sim \mathrm{Unif}\,[0, 10]$$
$$\mathrm{dir} \sim \mathrm{unif.\ over\ sphere}$$

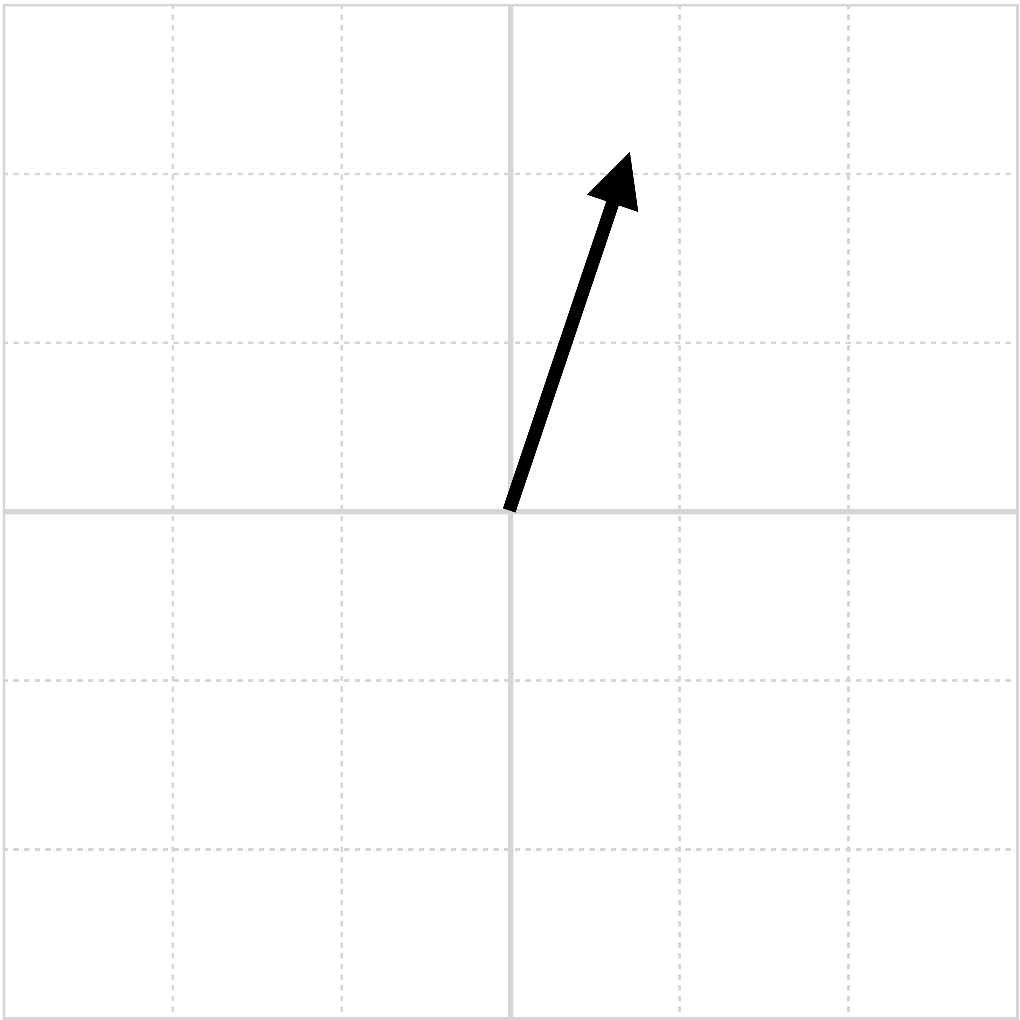
$p_{\mathrm{edit}}\left(z_e\right)$



y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

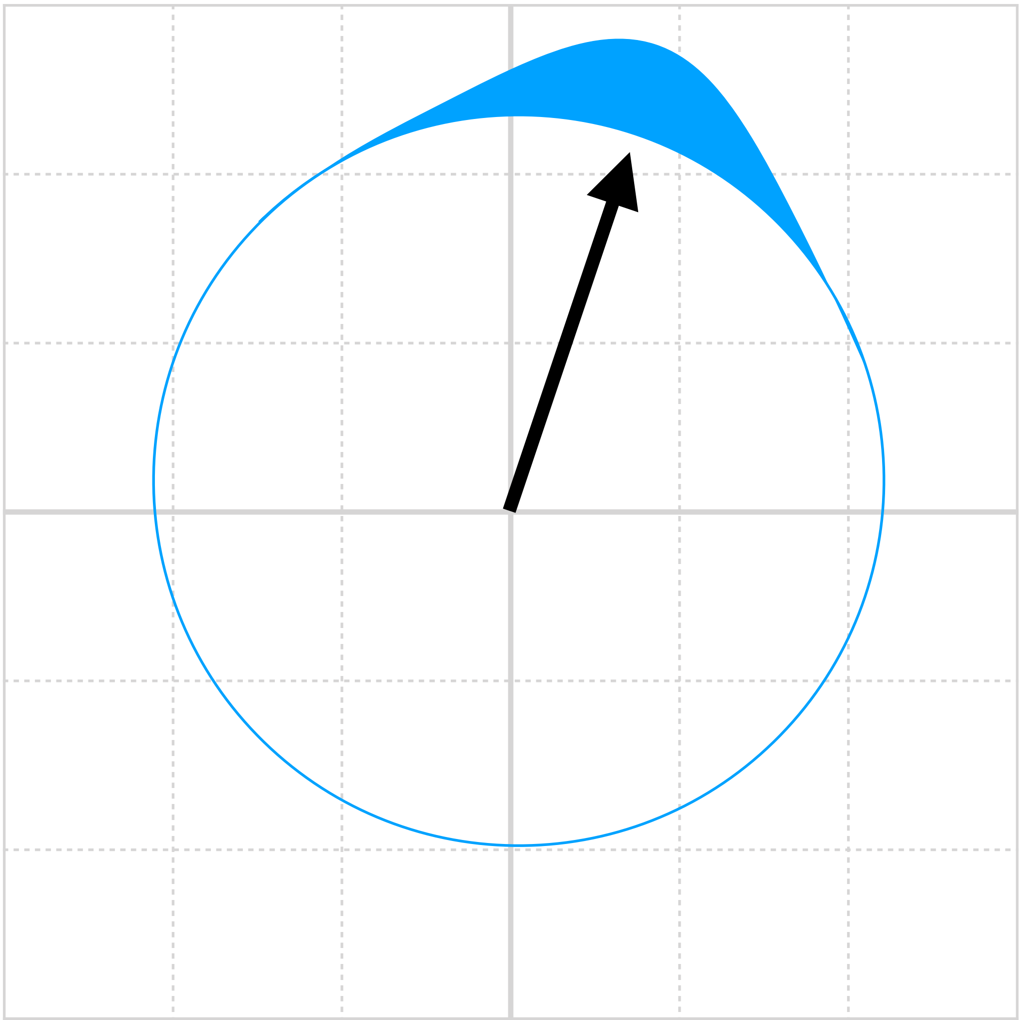
How to add noise to $\widehat{z_e}$?

$$\widehat{z_e}$$



How to add noise to $\widehat{z_e}$?

$\widehat{z_e}$

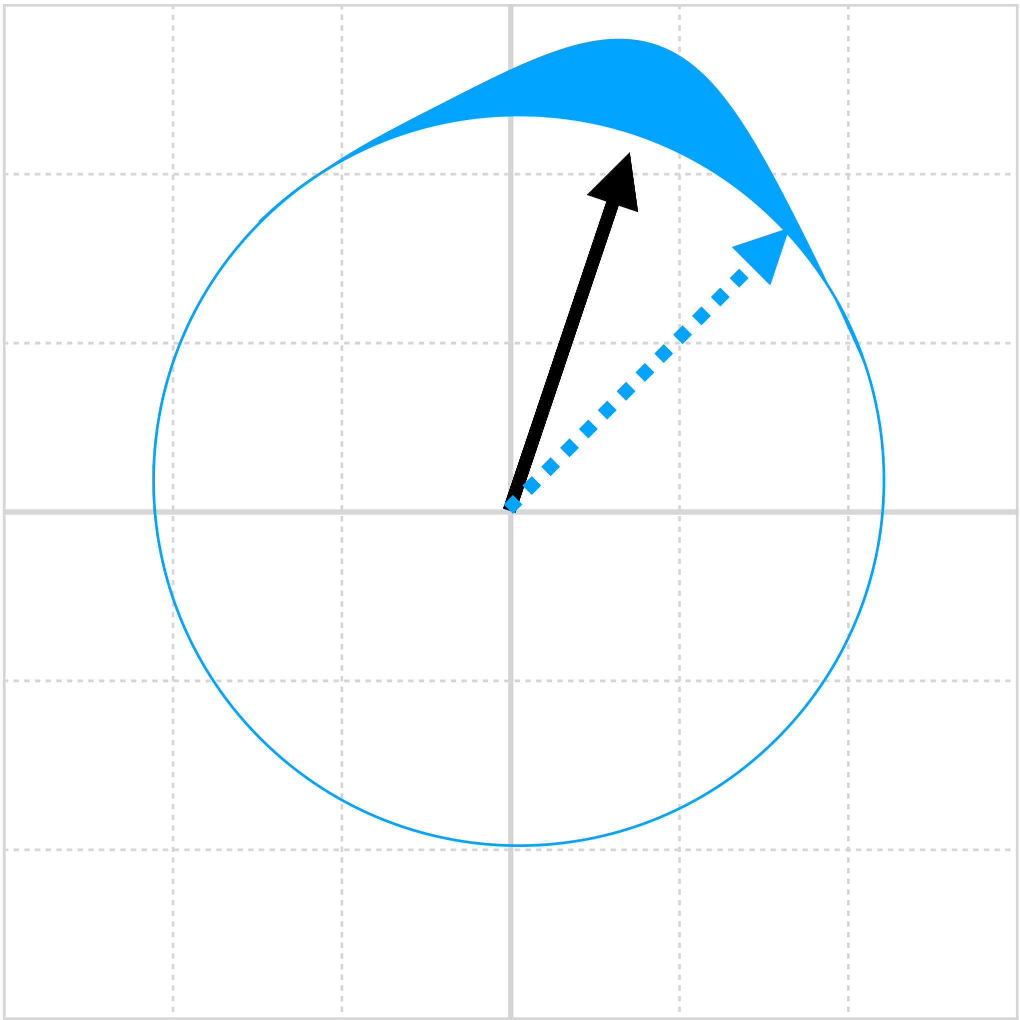


random rotation

von Mises-Fisher distribution

How to add noise to $\widehat{z_e}$?

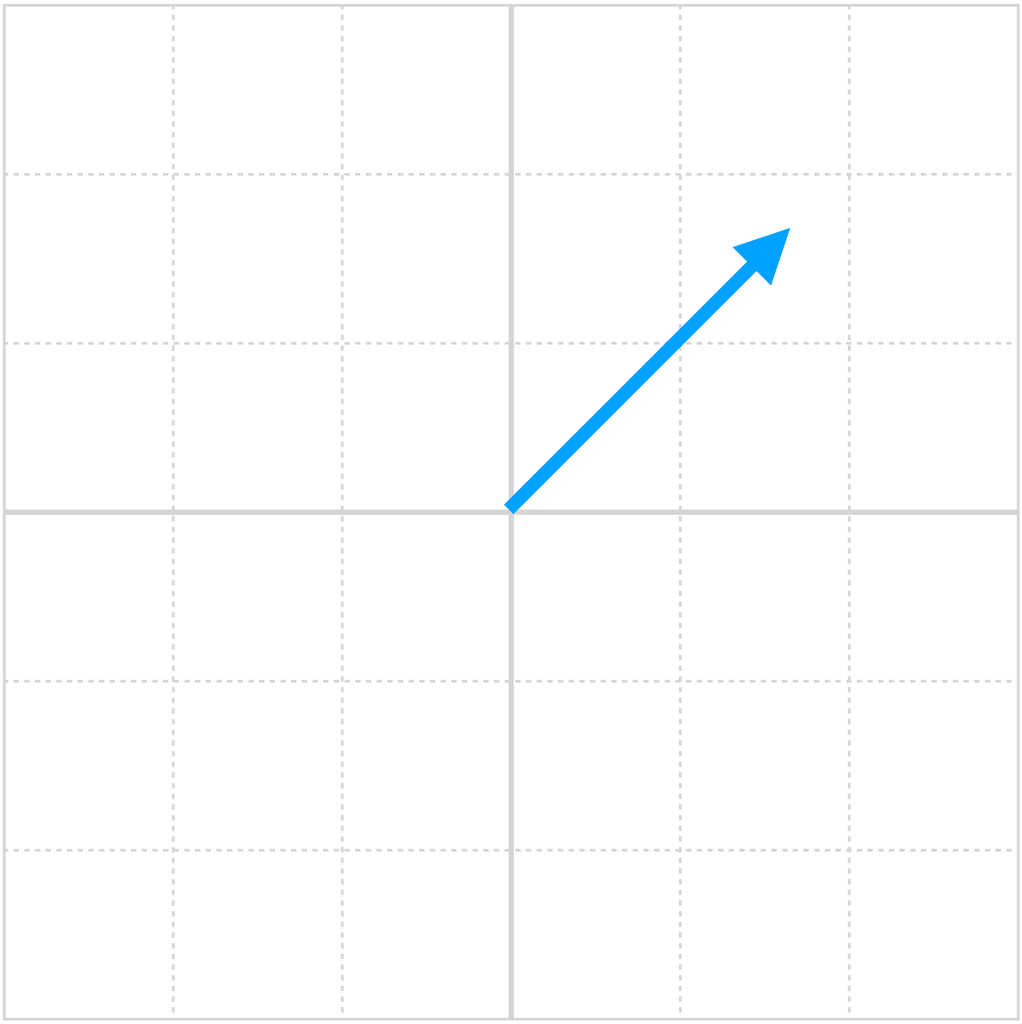
$\widehat{z_e}$



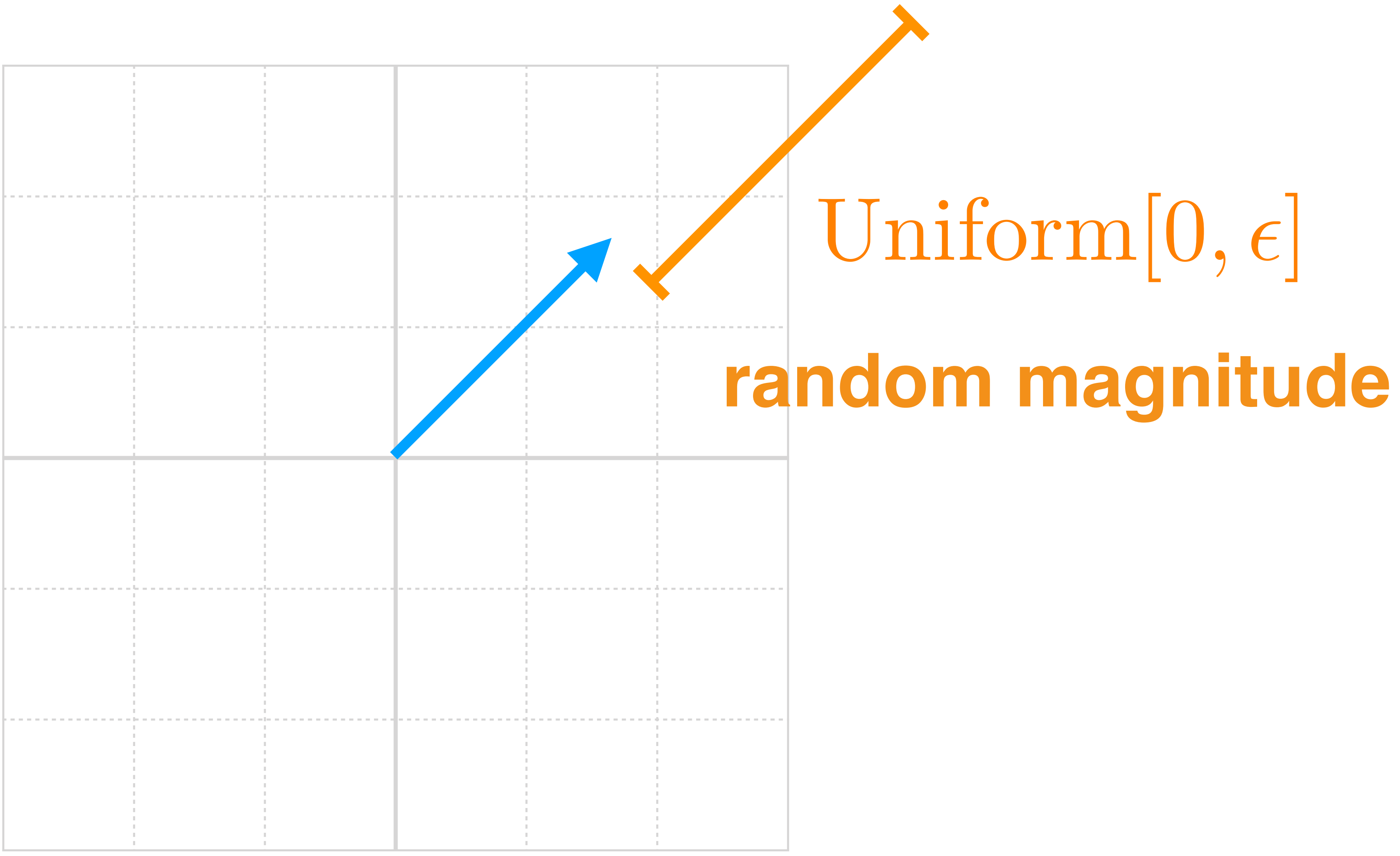
random rotation

von Mises-Fisher distribution

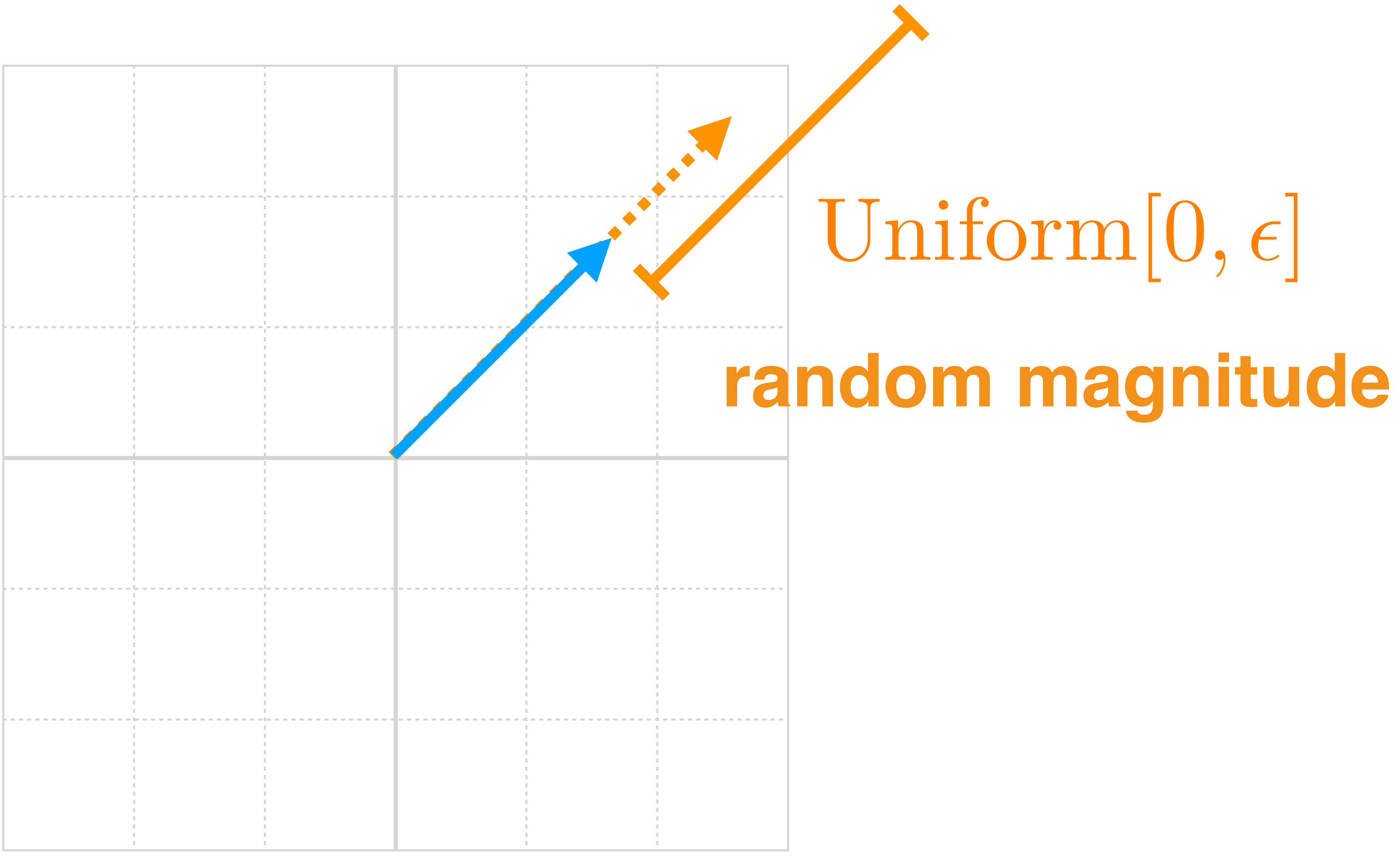
How to add noise to $\widehat{z_e}$?



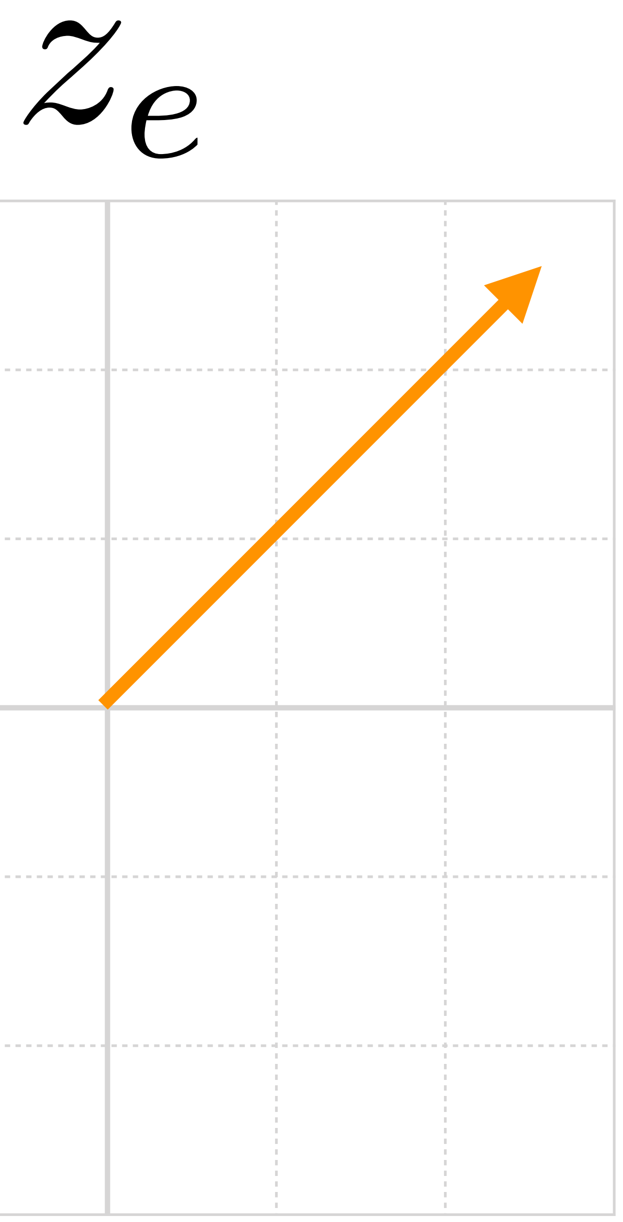
How to add noise to $\widehat{z_e}$?



How to add noise to $\widehat{z_e}$?

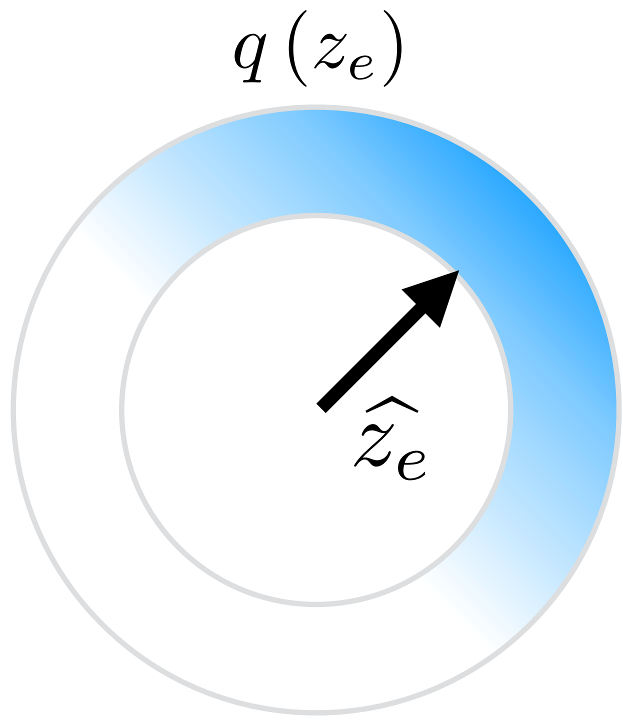


How to add noise to $\widehat{z_e}$?

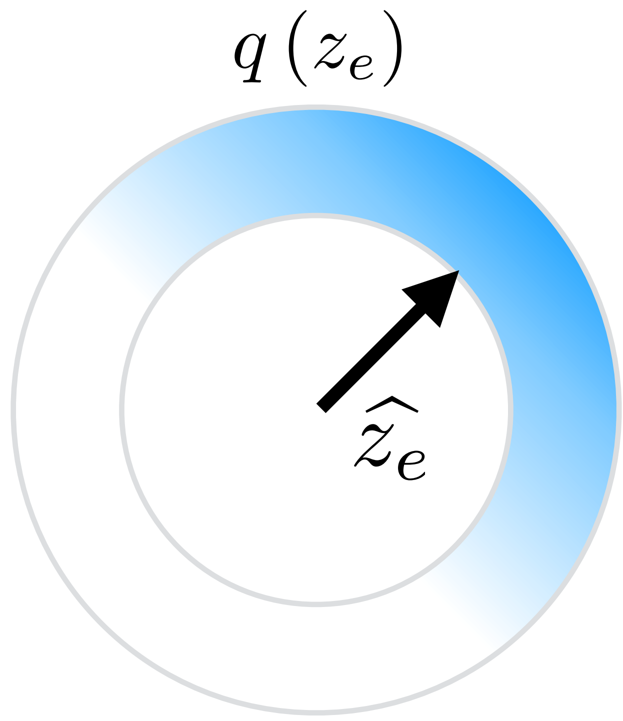


q(z) over edits

q(z) over edits

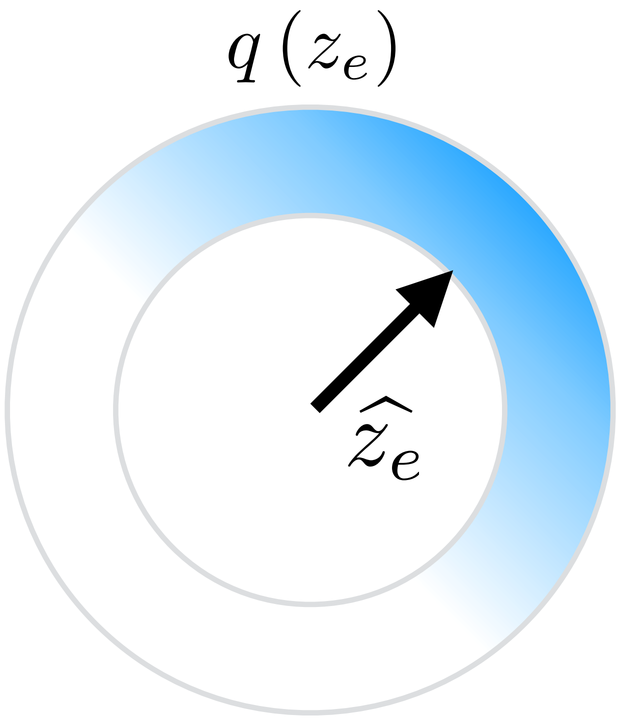


q(z) over edits



$$\mathrm{dir} \sim \mathrm{vMF}\left(\widehat{\mathrm{dir}}, \kappa\right)$$

q(z) over edits

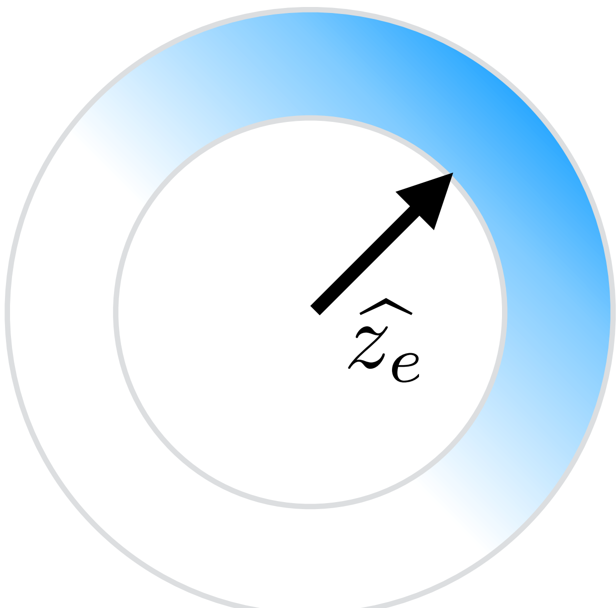


$$\mathrm{dir} \sim \mathrm{vMF}\left(\widehat{\mathrm{dir}}, \kappa\right)$$

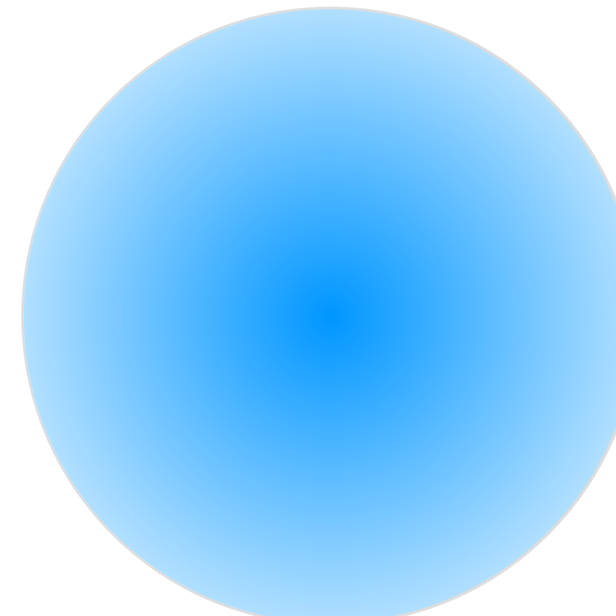
$$\mathrm{mag} \sim \mathrm{Unif}\left[\widehat{\mathrm{mag}}, \widehat{\mathrm{mag}} + \epsilon\right]$$

q(z) over edits

$q\left(z_e\right)$



$p_{\text{edit}}\left(z_e\right)$

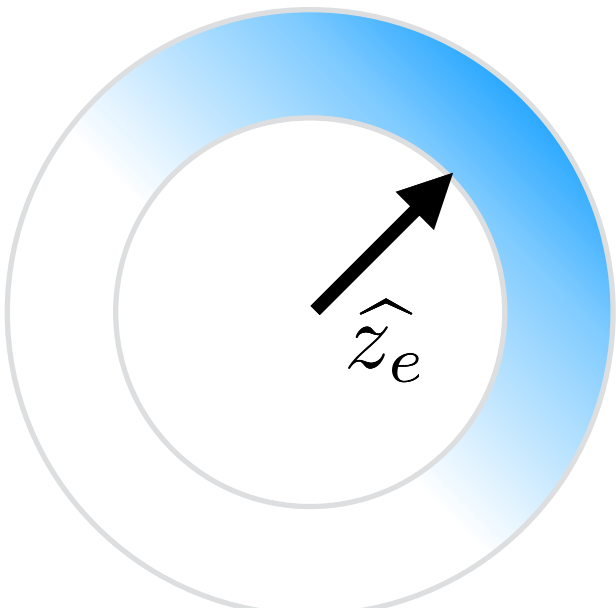


$$\text{dir} \sim \text{vMF}\left(\widehat{\text{dir}}, \kappa\right)$$

$$\text{mag} \sim \text{Unif}\left[\widehat{\text{mag}}, \widehat{\text{mag}} + \epsilon\right]$$

q(z) over edits

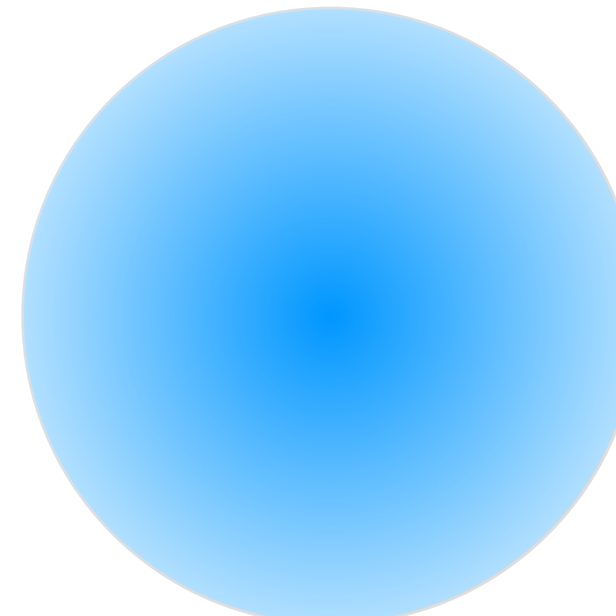
$q\left(z_e\right)$



$$\mathrm{dir} \sim \mathrm{vMF}\left(\widehat{\mathrm{dir}}, \kappa\right)$$

$$\mathrm{mag} \sim \mathrm{Unif}\left[\widehat{\mathrm{mag}}, \widehat{\mathrm{mag}} + \epsilon\right]$$

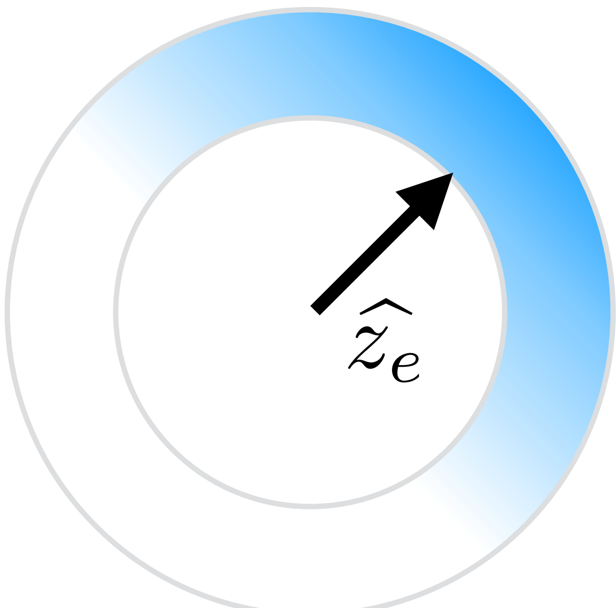
$p_{\mathrm{edit}}\left(z_e\right)$



$$\mathrm{dir} \sim \text{unif. over sphere}$$

q(z) over edits

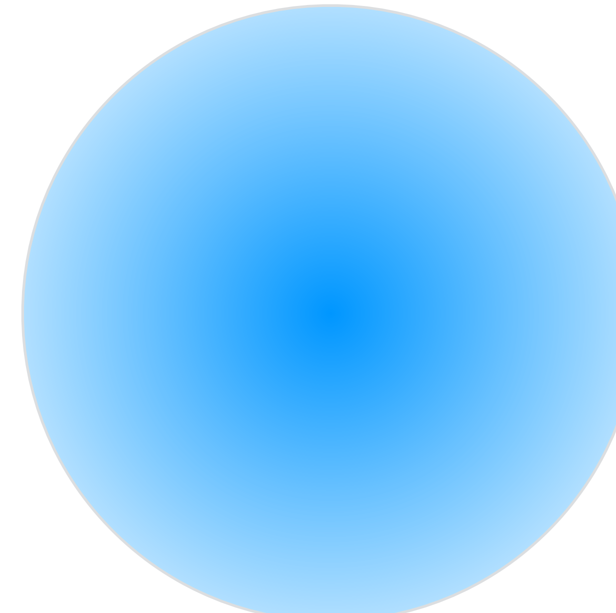
$q\left(z_e\right)$



$$\text{dir} \sim \text{vMF}\left(\widehat{\text{dir}}, \kappa\right)$$

$$\text{mag} \sim \text{Unif}\left[\widehat{\text{mag}}, \widehat{\text{mag}} + \epsilon\right]$$

$p_{\text{edit}}\left(z_e\right)$

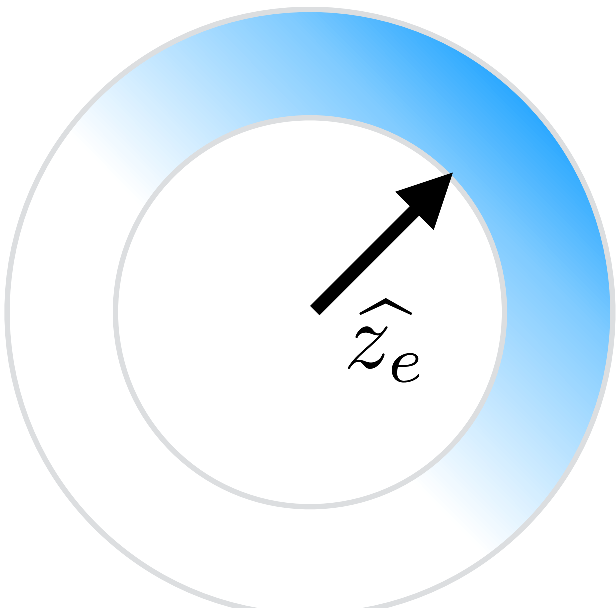


$$\text{dir} \sim \text{unif. over sphere}$$

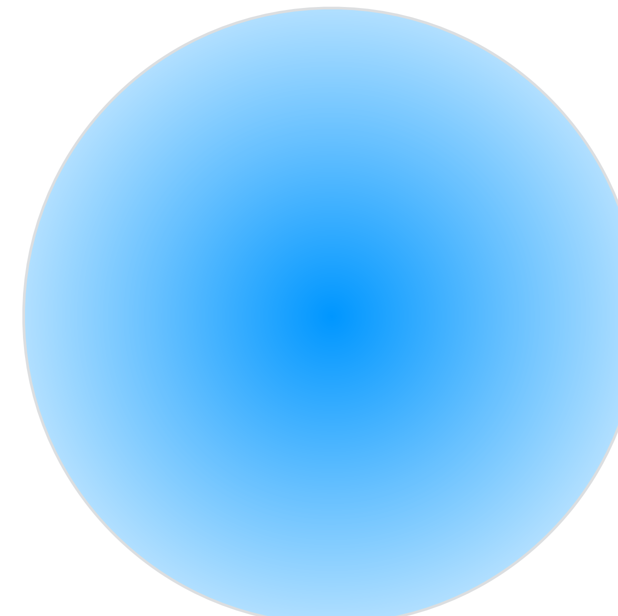
$$\text{mag} \sim \text{Unif}\left[0, 10\right]$$

q(z) over edits

$q\left(z_e\right)$



$p_{\text{edit}}\left(z_e\right)$



$$\text{dir} \sim \text{vMF}\left(\widehat{\text{dir}}, \kappa\right)$$

$$\text{mag} \sim \text{Unif}\left[\widehat{\text{mag}}, \widehat{\text{mag}} + \epsilon\right]$$

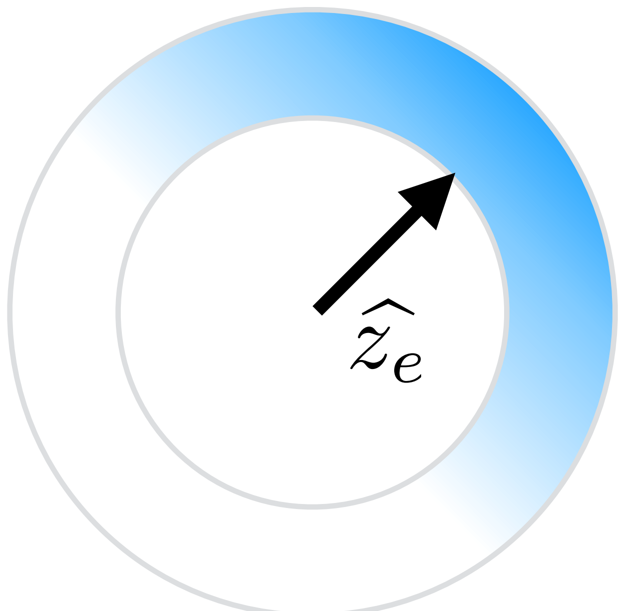
$$\text{dir} \sim \text{unif. over sphere}$$

$$\text{mag} \sim \text{Unif}\left[0, 10\right]$$

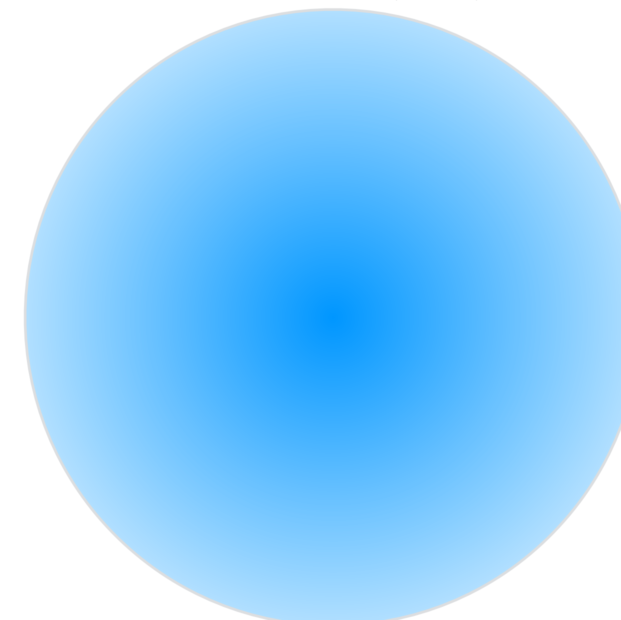
ELBO = reconstruction_cost - KL_penalty

q(z) over edits

$q\left(z_e\right)$



$p_{\text{edit}}\left(z_e\right)$



$$\text{dir} \sim \text{vMF}\left(\widehat{\text{dir}}, \kappa\right)$$

$$\text{mag} \sim \text{Unif}\left[\widehat{\text{mag}}, \widehat{\text{mag}} + \epsilon\right]$$

$$\text{dir} \sim \text{unif. over sphere}$$

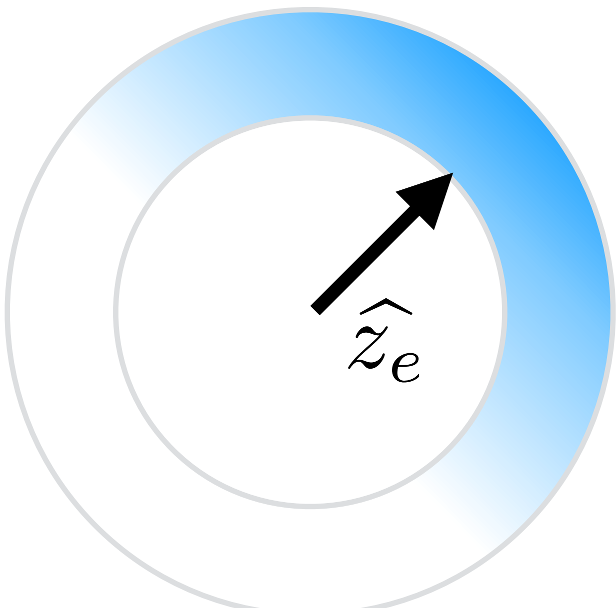
$$\text{mag} \sim \text{Unif}\left[0, 10\right]$$

ELBO = reconstruction_cost - KL_penalty

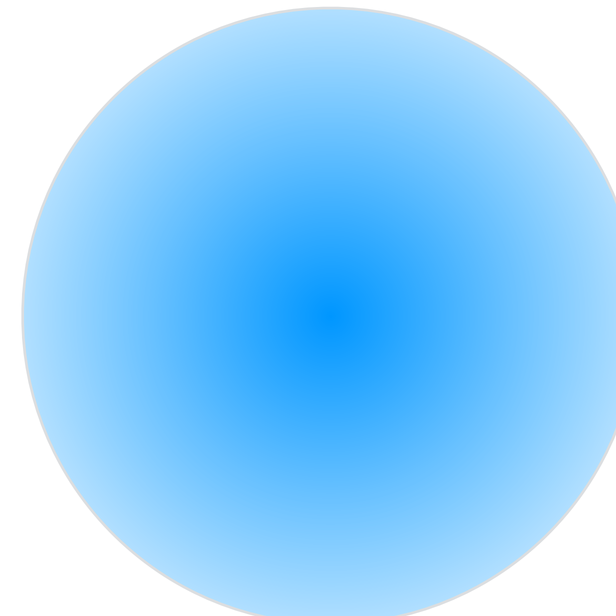
reparameterization trick (VAEs)

q(z) over edits

$q\left(z_e\right)$



$p_{\text{edit}}\left(z_e\right)$



$\text{dir} \sim \text{vMF}\left(\widehat{\text{dir}}, \kappa\right)$

$\text{mag} \sim \text{Unif}\left[\widehat{\text{mag}}, \widehat{\text{mag}} + \epsilon\right]$

$\text{dir} \sim \text{unif. over sphere}$

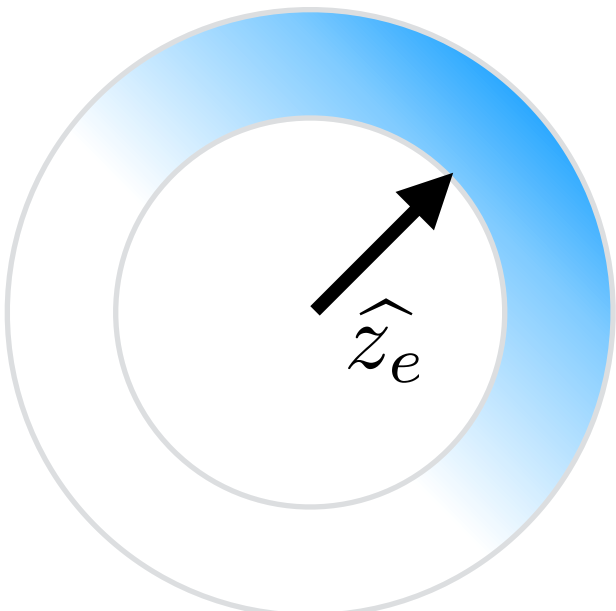
$\text{mag} \sim \text{Unif}\left[0, 10\right]$

ELBO = reconstruction_cost - KL_penalty

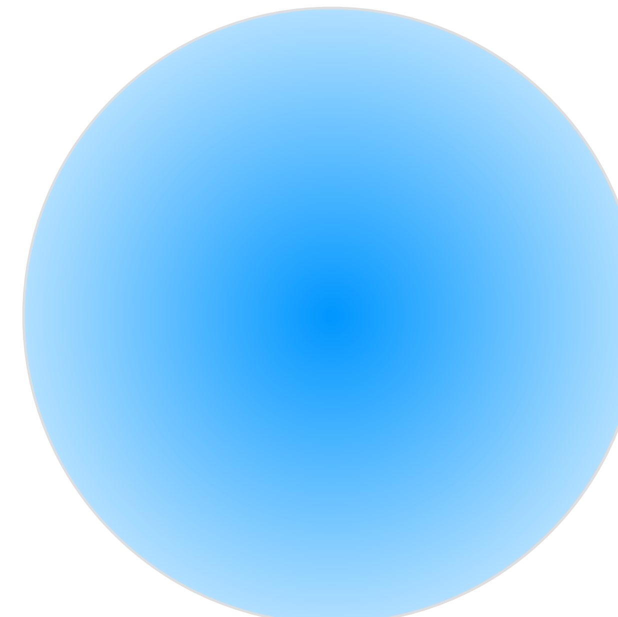
reparameterization trick (VAEs) **just a constant**

q(z) over edits

$q\left(z_e\right)$



$p_{\text{edit}}\left(z_e\right)$



$$\text{dir} \sim \text{vMF}\left(\widehat{\text{dir}}, \kappa\right)$$

$$\text{mag} \sim \text{Unif}\left[\widehat{\text{mag}}, \widehat{\text{mag}} + \epsilon\right]$$

$$\text{dir} \sim \text{unif. over sphere}$$

$$\text{mag} \sim \text{Unif}\left[0, 10\right]$$

ELBO = reconstruction_cost - KL_penalty

reparameterization trick (VAEs)

just a constant

✓ **computationally tractable**

Summary of training

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Summary of training

- Build a training set of lexically similar sentence pairs ($\mathbf{z}_p$, $\mathbf{y}$)

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Summary of training

- Build a training set of lexically similar sentence pairs ($\mathbf{z}_p$, $\mathbf{y}$)

- For each pair of sentences ($\mathbf{z}_p$, $\mathbf{y}$)

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Summary of training

- Build a training set of lexically similar sentence pairs ($\mathbf{z}_p$, $\mathbf{y}$)
- For each pair of sentences ($\mathbf{z}_p$, $\mathbf{y}$)
    1. identify words that differ between $\mathbf{z}_p$ and $\mathbf{y}$

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Summary of training

- Build a training set of lexically similar sentence pairs ($\mathbf{z}_p$, $\mathbf{y}$)
- For each pair of sentences ($\mathbf{z}_p$, $\mathbf{y}$)
    1. identify words that differ between $\mathbf{z}_p$ and $\mathbf{y}$
    2. embed those words into a vector

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Summary of training

- Build a training set of lexically similar sentence pairs ($\mathbf{z}_p$, $\mathbf{y}$)
- For each pair of sentences ($\mathbf{z}_p$, $\mathbf{y}$)
  1. identify words that differ between $\mathbf{z}_p$ and $\mathbf{y}$
  2. embed those words into a vector
  3. add noise to get edit vector $\mathbf{z}_e$

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Summary of training

- Build a training set of lexically similar sentence pairs ($\mathbf{z}_p$, $\mathbf{y}$)
- For each pair of sentences ($\mathbf{z}_p$, $\mathbf{y}$)

1. identify words that differ between $\mathbf{z}_p$ and $\mathbf{y}$
2. embed those words into a vector
3. add noise to get edit vector $\mathbf{z}_e$
4. train seq2seq mapping ($\mathbf{z}_p$, $\mathbf{z}_e$) —> $\mathbf{y}$ $\quad p_{\text{editor}}(y \mid z_p, z_e)$

$\mathbf{y}$ = output sentence $\quad$ $\mathbf{z}_p$ = prototype sentence $\quad$ $\mathbf{z}_e$ = edit vector

Summary of training

- Build a training set of lexically similar sentence pairs ($\mathbf{z}_p$, $\mathbf{y}$)
- For each pair of sentences ($\mathbf{z}_p$, $\mathbf{y}$)

1. identify words that differ between $\mathbf{z}_p$ and $\mathbf{y}$
2. embed those words into a vector
3. add noise to get edit vector $\mathbf{z}_e$
4. train seq2seq mapping ($\mathbf{z}_p$, $\mathbf{z}_e$) —> $\mathbf{y}$ $\quad p_{\text{editor}}(y \mid z_p, z_e)$
5. update $\mathbf{q(z_e)}$

$\mathbf{y}$ = output sentence $\quad \mathbf{z}_p$ = prototype sentence $\quad \mathbf{z}_e$ = edit vector

\end{**Approach**}

\begin{**Results**}

i had the fried whitefish taco which was decent, but i've had much better.	i had the <unk> and the fried carnitas tacos, it was pretty tasty, but i've had better.
"hash browns" are unseasoned, frozen potato shreds burnt to a crisp on the outside and mushy on the inside.	the hash browns were crispy on the outside, but still the taste was missing.
i'm not sure what is preventing me from giving it <cardinal> stars, but i probably should.	i'm currently giving <cardinal> stars for the service alone.
quick place to grab light and tasty teriyaki.	this place is good and a quick place to grab a tasty sandwich.
sad part is we've been there before and its been good.	i've been here several times and always have a good time.

Prototype $\mathbf{z}_p$

i had the fried whitefish taco which was decent, but i've had much better.	i had the <unk> and the fried carnitas tacos, it was pretty tasty, but i've had better.
"hash browns" are unseasoned, frozen potato shreds burnt to a crisp on the outside and mushy on the inside.	the hash browns were crispy on the outside, but still the taste was missing.
i'm not sure what is preventing me from giving it <cardinal> stars, but i probably should.	i'm currently giving <cardinal> stars for the service alone.
quick place to grab light and tasty teriyaki.	this place is good and a quick place to grab a tasty sandwich.
sad part is we've been there before and its been good.	i've been here several times and always have a good time.

Prototype $\mathbf{z}_p$	Output $\mathbf{y}$
i had the fried whitefish taco which was decent, but i've had much better.	i had the <unk> and the fried carnitas tacos, it was pretty tasty, but i've had better.
"hash browns" are unseasoned, frozen potato shreds burnt to a crisp on the outside and mushy on the inside.	the hash browns were crispy on the outside, but still the taste was missing.
i'm not sure what is preventing me from giving it <cardinal> stars, but i probably should.	i'm currently giving <cardinal> stars for the service alone.
quick place to grab light and tasty teriyaki.	this place is good and a quick place to grab a tasty sandwich.
sad part is we've been there before and its been good.	i've been here several times and always have a good time.

Prototype z_p (random edit vector) **Output y**

i had the fried whitefish taco which was decent, but i've had much better.	i had the <unk> and the fried carnitas tacos, it was pretty tasty, but i've had better.
"hash browns" are unseasoned, frozen potato shreds burnt to a crisp on the outside and mushy on the inside.	the hash browns were crispy on the outside, but still the taste was missing.
i'm not sure what is preventing me from giving it <cardinal> stars, but i probably should.	i'm currently giving <cardinal> stars for the service alone.
quick place to grab light and tasty teriyaki.	this place is good and a quick place to grab a tasty sandwich.
sad part is we've been there before and its been good.	i've been here several times and always have a good time.

Overview of results

- **More diverse generations**
- **Higher quality generations**
- **Better perplexity** (BillionWord, Yelp reviews)
- **Edits are semantically meaningful**
  - preserve semantic similarity
  - can be used to perform sentence-level analogies

Overview of results

- **More diverse generations**
- **Higher quality generations**
- **Better perplexity** (BillionWord, Yelp reviews)
- ✓ **Edits are semantically meaningful**
  - preserve semantic similarity
  - can be used to perform sentence-level analogies

Edits are semantically meaningful

$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Edits are semantically meaningful

$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

plug in your own edit vector!

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Edits are semantically meaningful

$$y \sim p_{\mathrm{editor}}\left(y \mid z_p, z_e\right)$$

plug in your own edit vector!

✓ **semantic control**

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Edits are semantically meaningful

$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

plug in your own edit vector!

✔ **semantic control**

semantic smoothness:

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Edits are semantically meaningful

$$y \sim p_{\mathrm{editor}}\left(y \mid z_p, z_e\right)$$

plug in your own edit vector!

✓ **semantic control**

semantic smoothness:
small magnitude edit vector should cause small changes

y = output sentence **z**$_p$ = prototype sentence **z**$_e$ = edit vector

Edits are semantically meaningful

$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

✓ **semantic control**

plug in your own edit vector!

semantic smoothness:
small magnitude edit vector should cause small changes

consistent edit behavior:

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Edits are semantically meaningful

$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

plug in your own edit vector!

✓ **semantic control**

semantic smoothness:
small magnitude edit vector should cause small changes

consistent edit behavior:
apply the same edit vector to different sentences

y = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Edits are semantically meaningful

$$y \sim p_{\text{editor}} \left(y \mid z_p, z_e \right)$$

✓ **semantic control**

plug in your own edit vector!

semantic smoothness:
small magnitude edit vector should cause small changes

consistent edit behavior:
apply the same edit vector to different sentences
should cause semantically analogous edits

$\mathbf{y}$ = output sentence $\mathbf{z}_p$ = prototype sentence $\mathbf{z}_e$ = edit vector

Semantic smoothness

Semantic smoothness

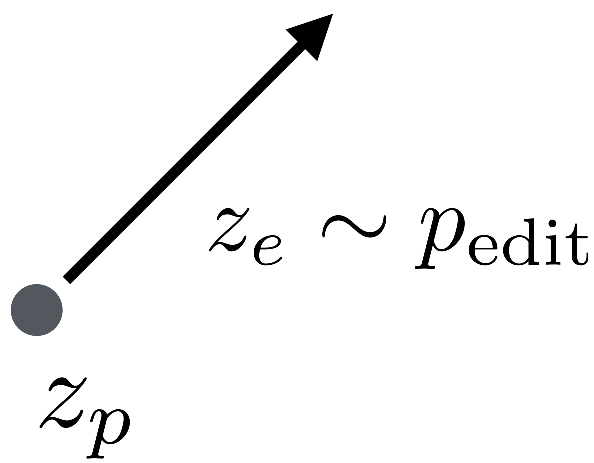
random walk in sentence space

Semantic smoothness

random walk in sentence space

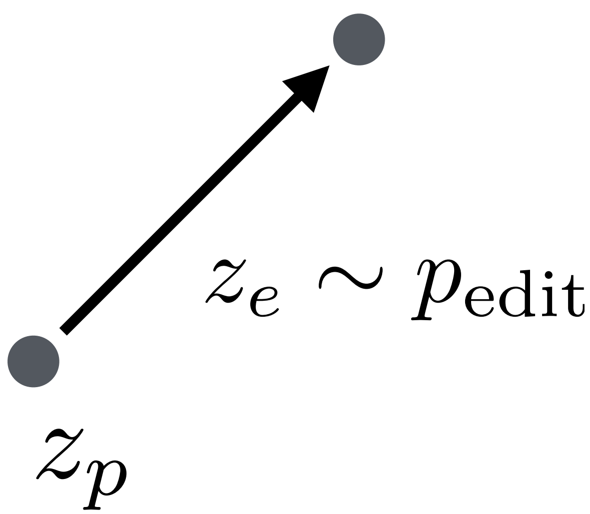
z_p

Semantic smoothness



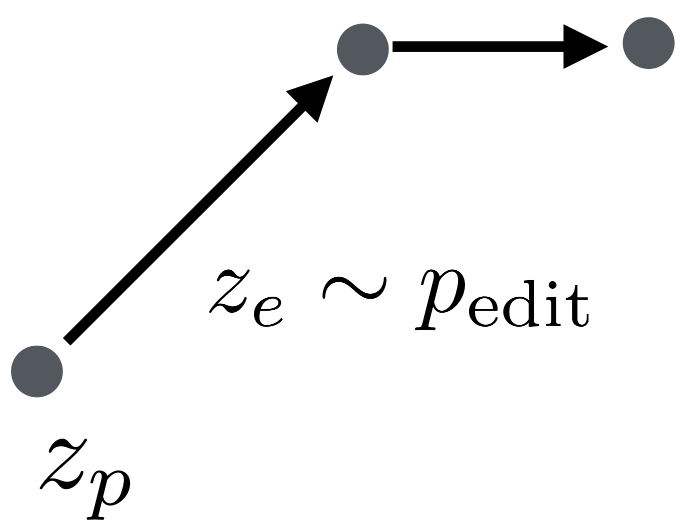
random walk in sentence space

Semantic smoothness



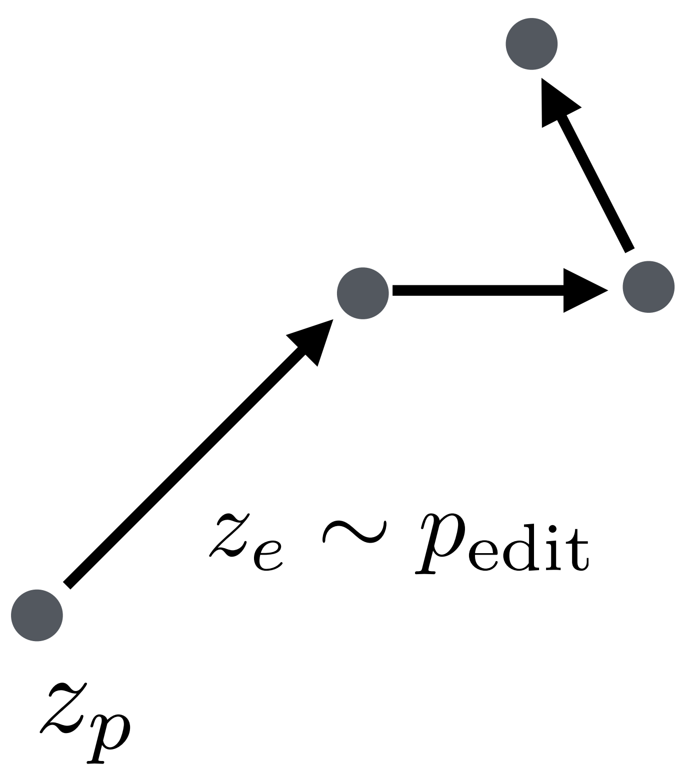
random walk in sentence space

Semantic smoothness



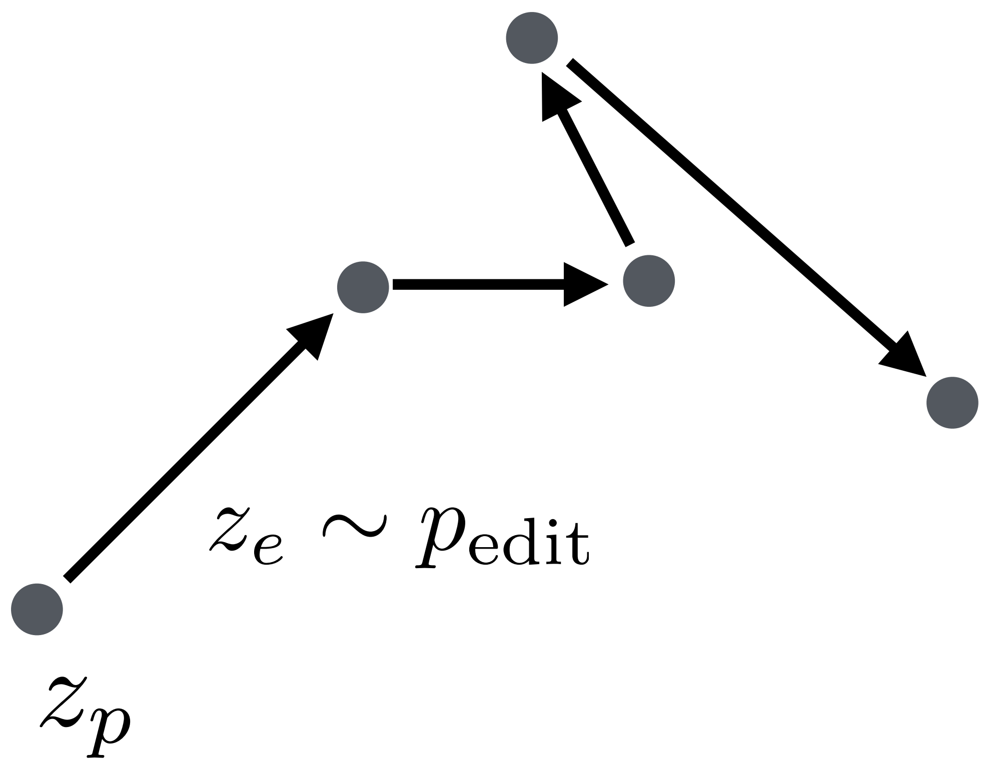
random walk in sentence space

Semantic smoothness



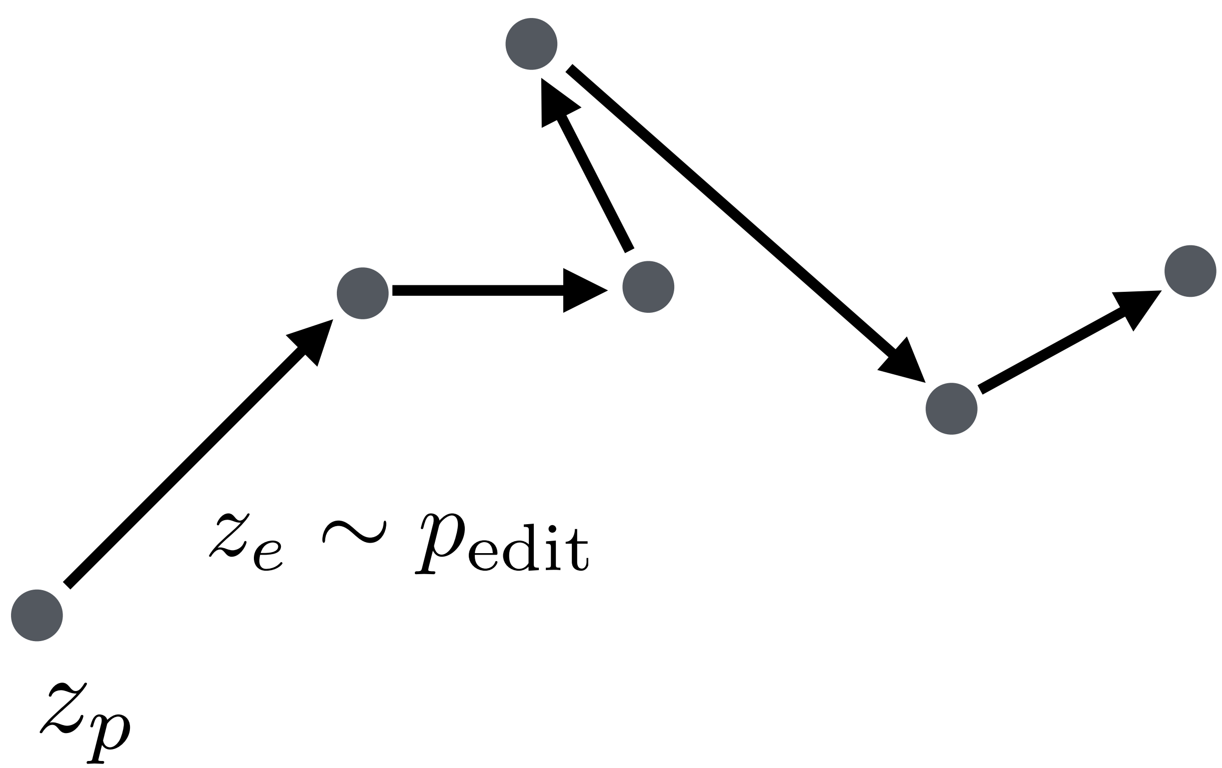
random walk in sentence space

Semantic smoothness



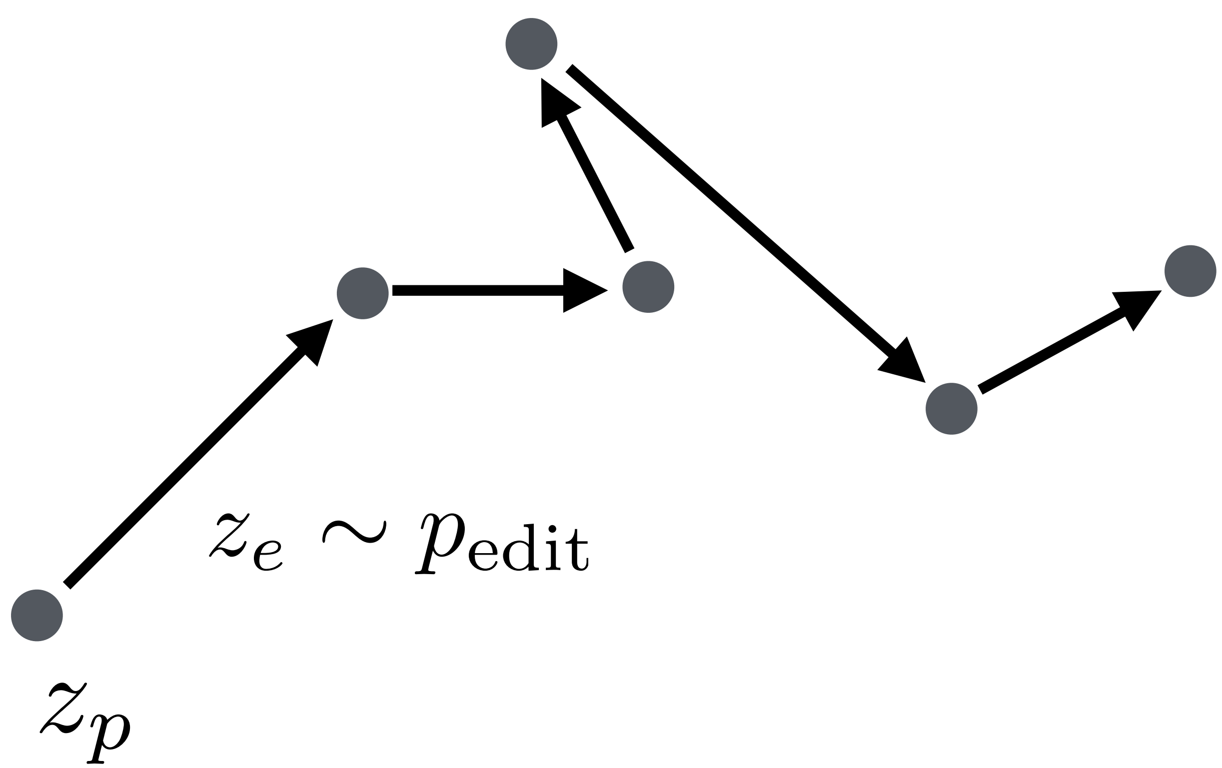
random walk in sentence space

Semantic smoothness



random walk in sentence space

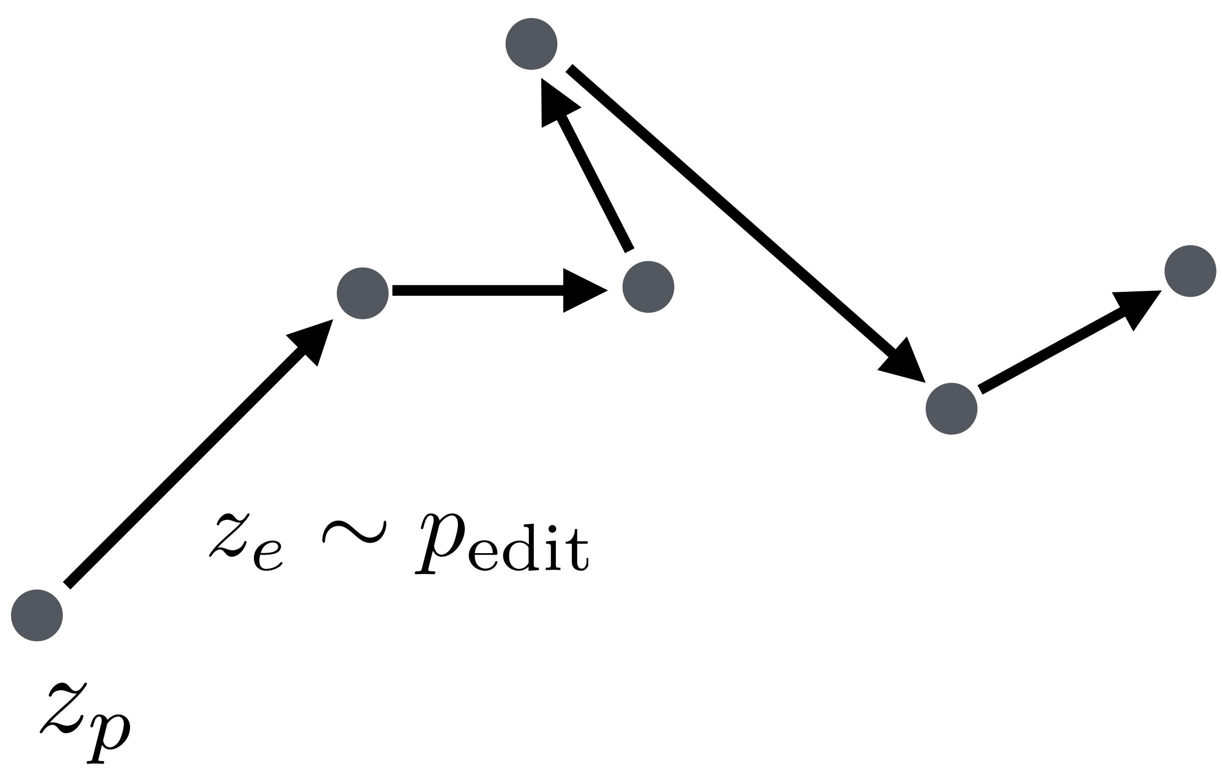
Semantic smoothness



random walk in sentence space

- `ice cream was one of the best i've ever tried .`

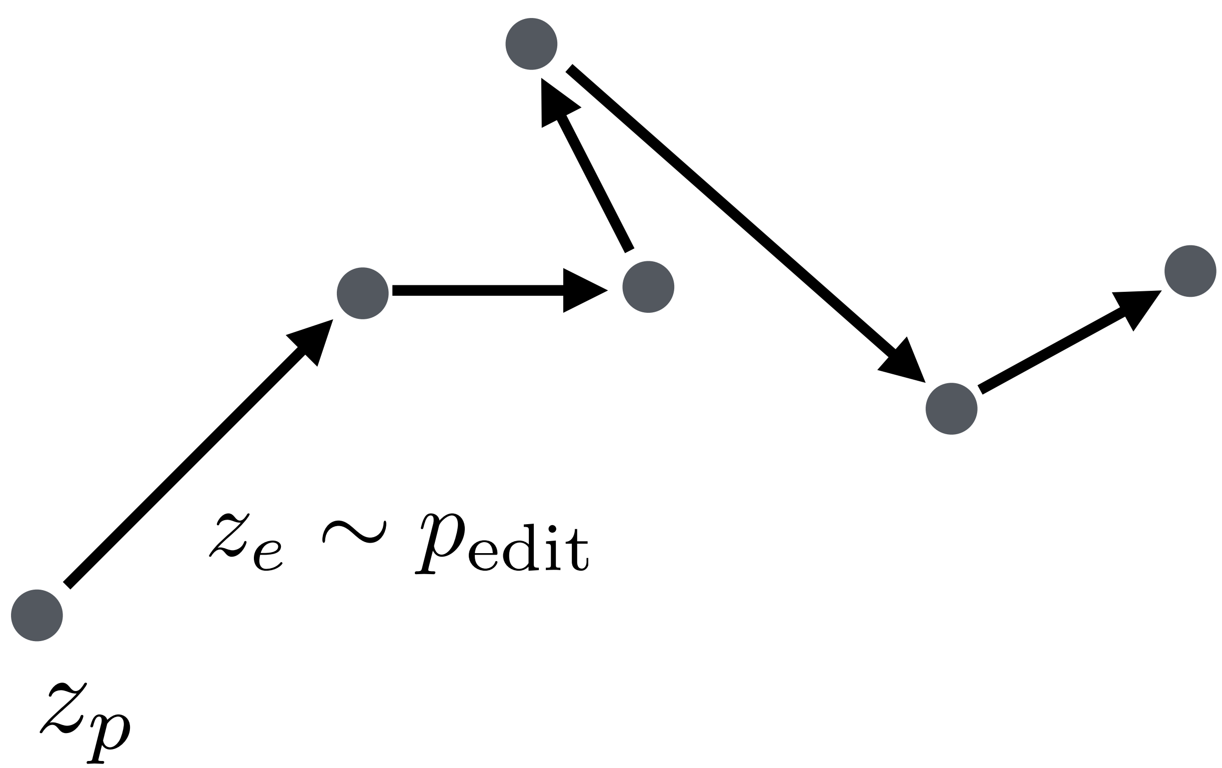
Semantic smoothness



random walk in sentence space

- `ice cream was one of the best i've ever tried .`
- `some of the best ice cream we've ever had .`

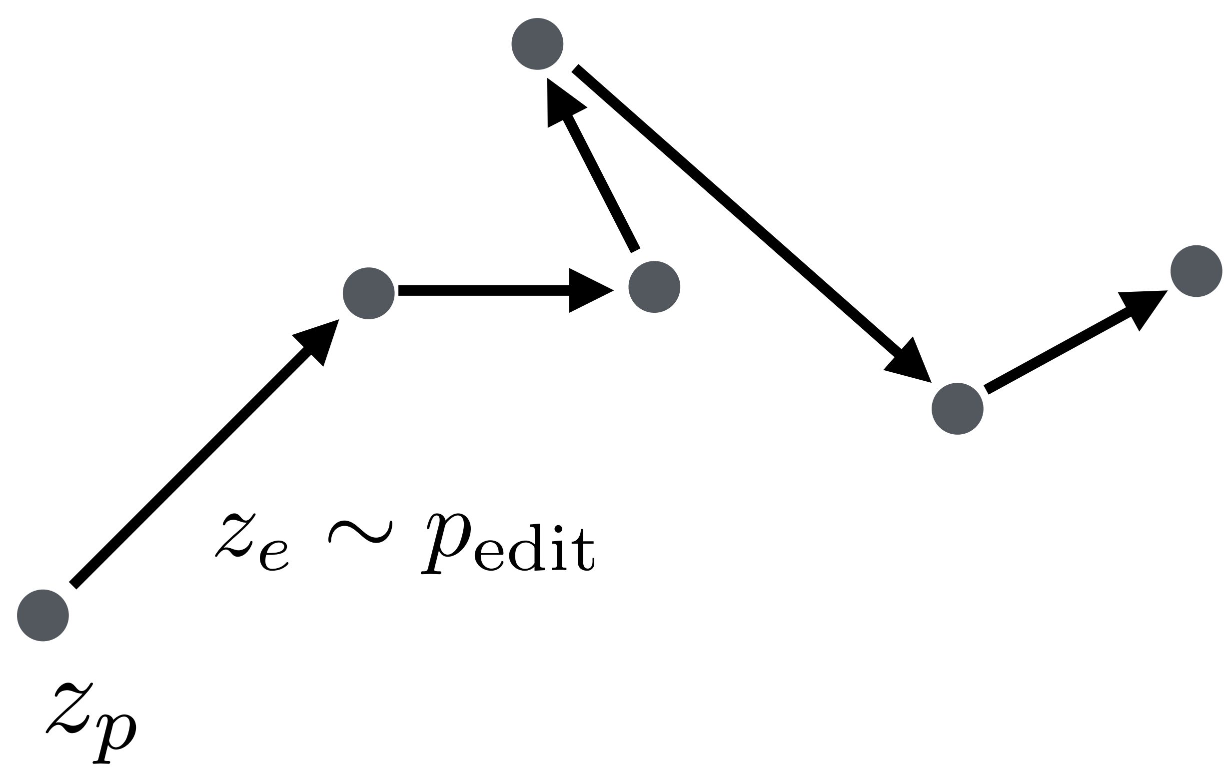
Semantic smoothness



random walk in sentence space

- `ice cream was one of the best i've ever tried .`
- `some of the best ice cream we've ever had .`
- `just had the best ice - cream i've ever had !`

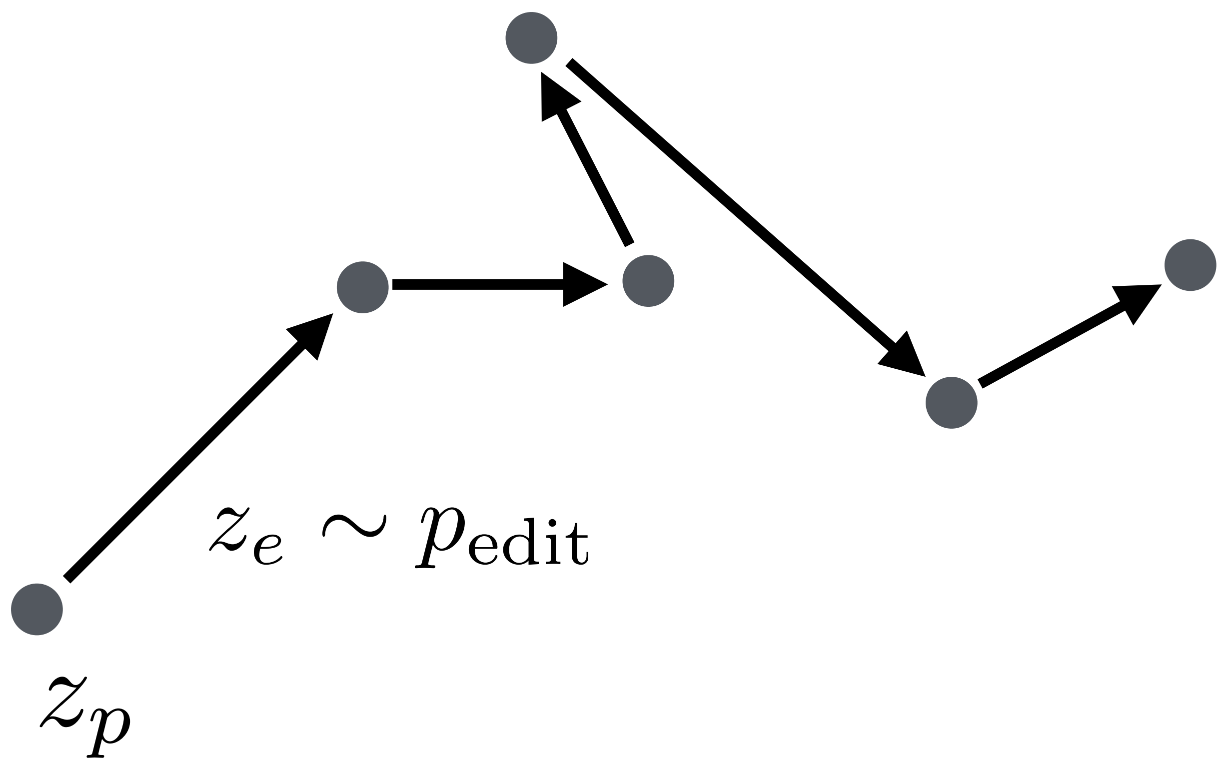
Semantic smoothness



random walk in sentence space

- ice cream was one of the best i've ever tried .
- some of the best ice cream we've ever had .
- just had the best ice - cream i've ever had !
- some of the best pizza i've ever tasted !

Semantic smoothness

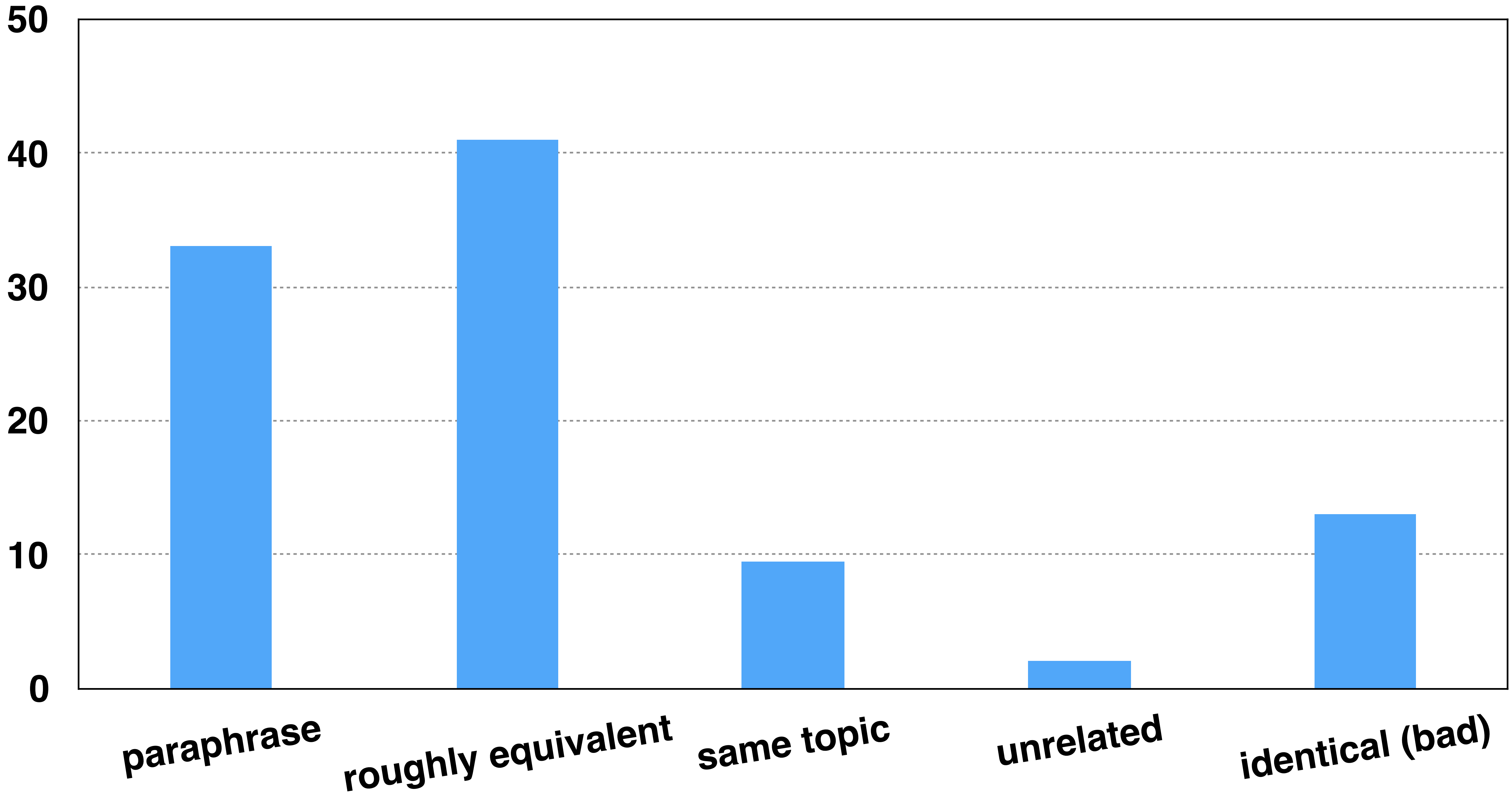


random walk in sentence space

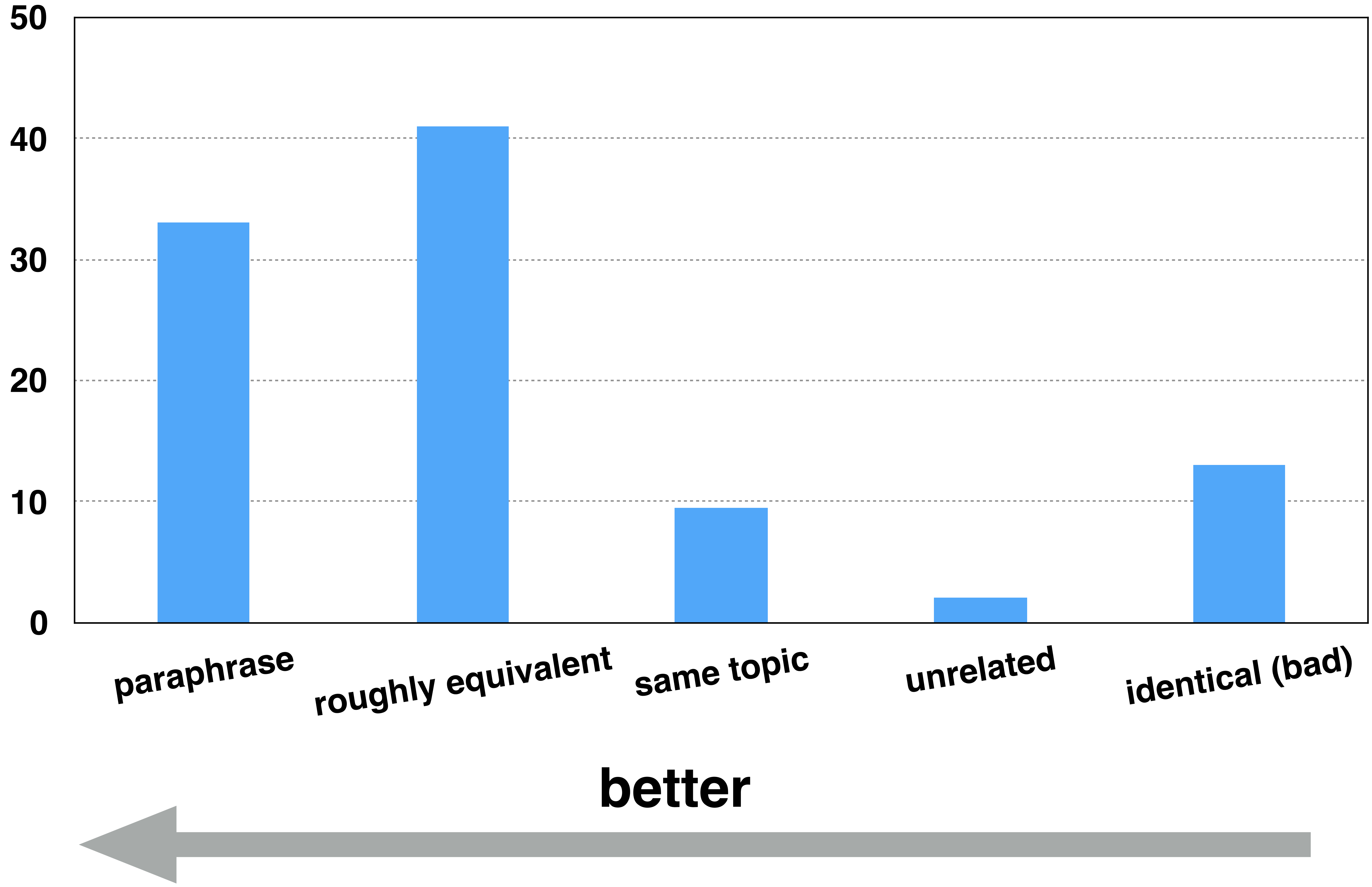
- `ice cream was one of the best i've ever tried .`
- `some of the best ice cream we've ever had .`
- `just had the best ice - cream i've ever had !`
- `some of the best pizza i've ever tasted !`
- `that was some of the best pizza i've had in the area .`

Turkers: how jumpy is each step?

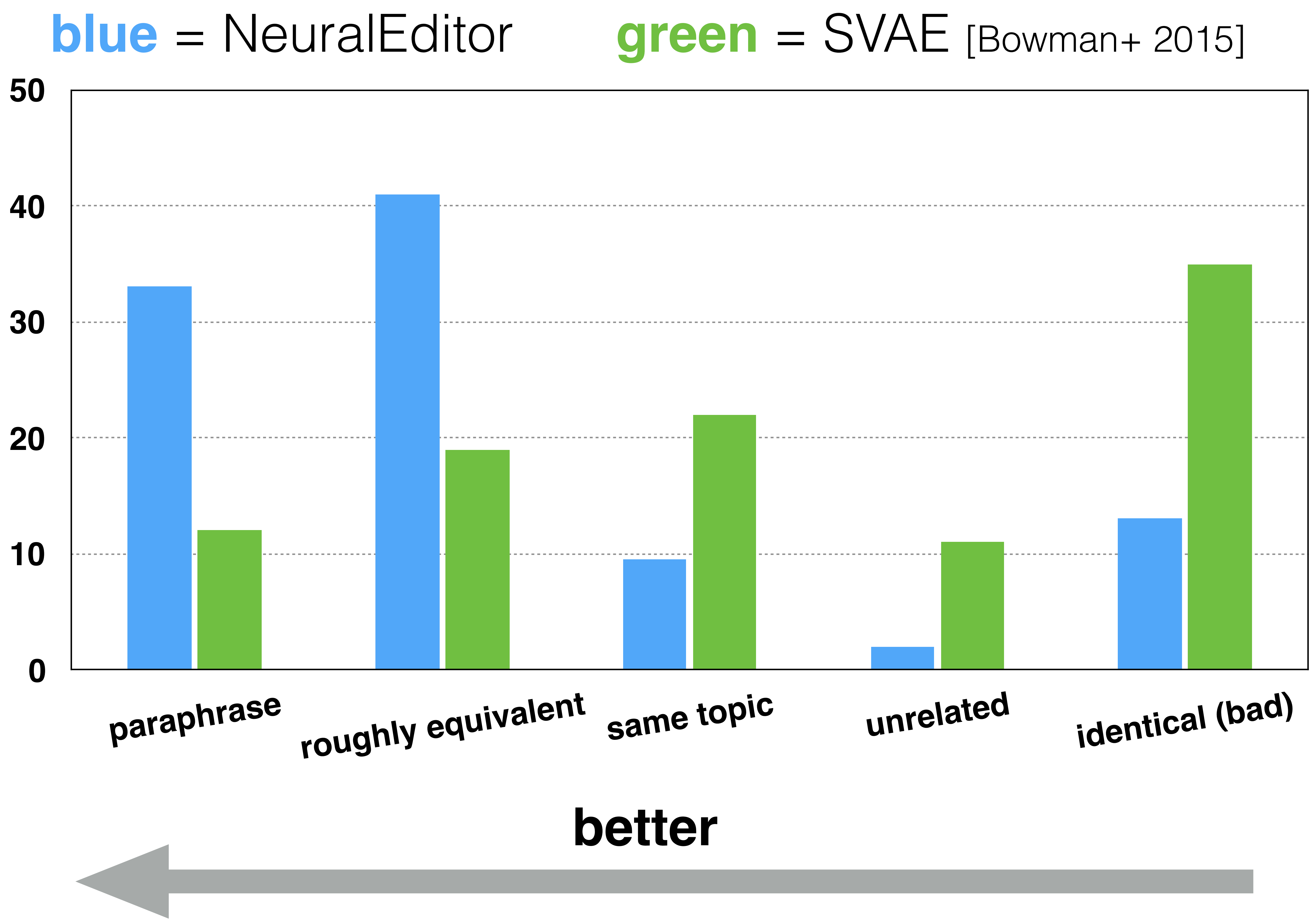
Turkers: how jumpy is each step?



Turkers: how jumpy is each step?



Turkers: how smooth is the random walk?

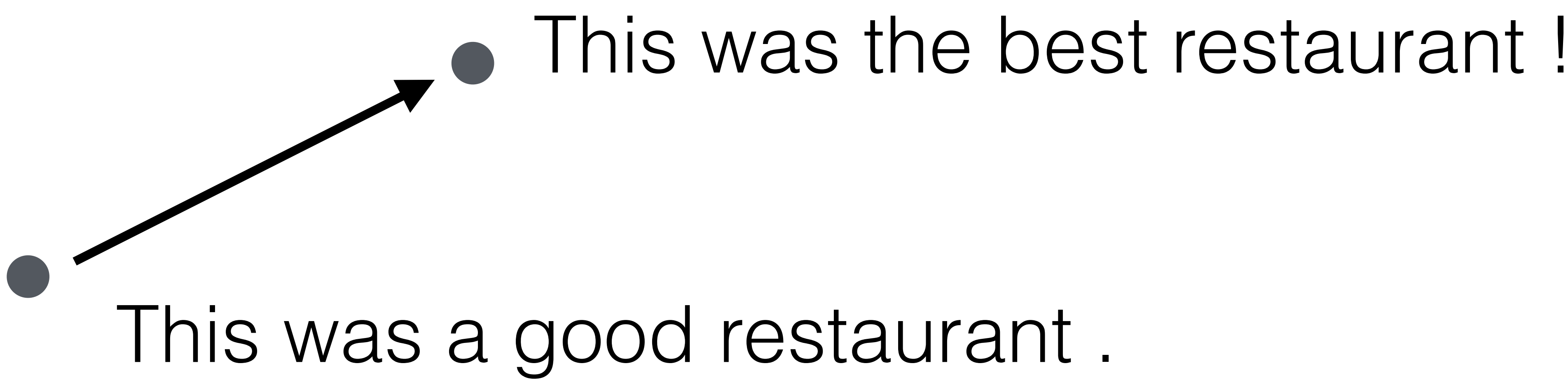


Consistent edit behavior

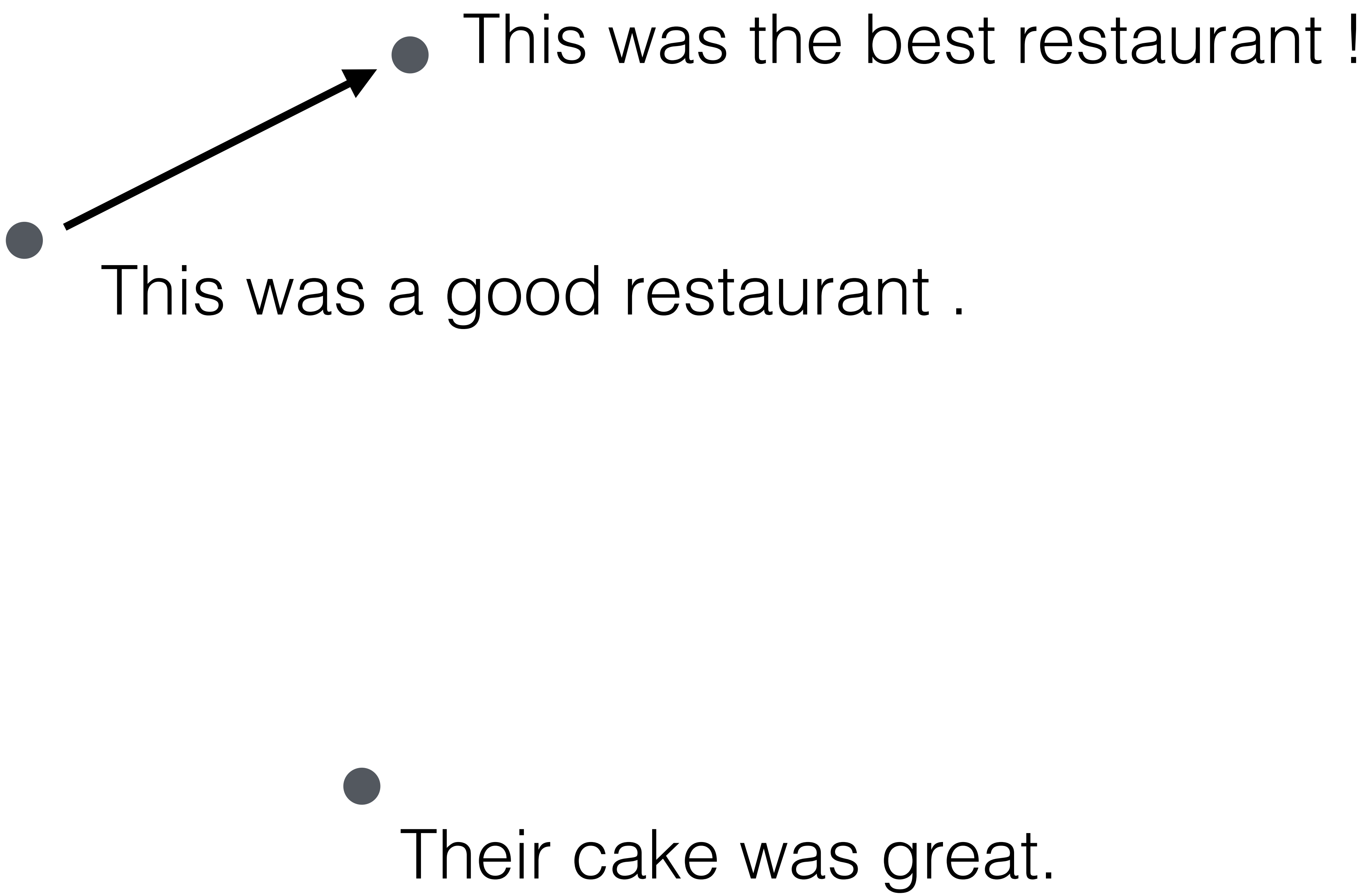
Consistent edit behavior

- This was a good restaurant .

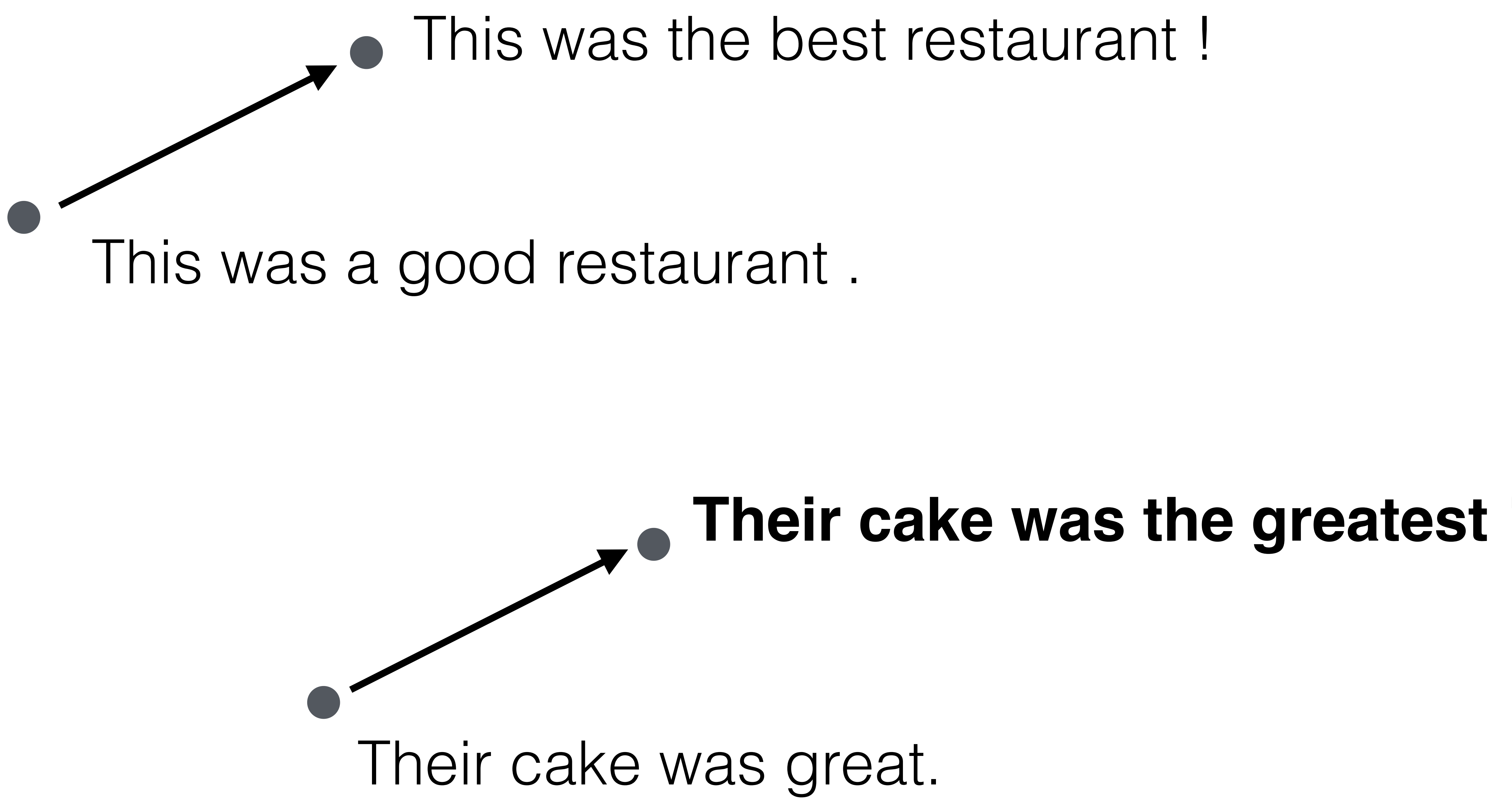
Consistent edit behavior



Consistent edit behavior



Consistent edit behavior



Sentence analogy dataset

Sentence analogy dataset

Lexical analogies [Mikolov+ 2013]

Sentence analogy dataset

Lexical analogies [Mikolov+ 2013]

is $\xrightarrow{\text{past tense}}$ **was**

Sentence analogy dataset

Lexical analogies [Mikolov+ 2013]

is $\xrightarrow{\text{past tense}}$ **was**

comes $\xrightarrow{\text{past tense}}$ **came**

Sentence analogy dataset

Lexical analogies [Mikolov+ 2013]

past tense

is → **was**

This **is** the place to go.

This **was** the place to go.

past tense

comes → **came**

Sentence analogy dataset

Lexical analogies [Mikolov+ 2013]

is $\xrightarrow{\text{past tense}}$ **was**

This **is** the place to go. This **was** the place to go.

comes $\xrightarrow{\text{past tense}}$ **came**

He **comes** home tired and happy. He **came** home happy and tired.

Sentence analogy dataset

Lexical analogies [Mikolov+ 2013]

is $\xrightarrow{\text{past tense}}$ **was**

This **is** the place to go. This **was** the place to go.

comes $\xrightarrow{\text{past tense}}$ **came**

He **comes** home tired and happy. He **came** home happy and tired.

(allow reordering and stopwords)

Sentence analogy dataset

This **is** the place to go. This **was** the place to go.

He **comes** home tired and happy. He **came** home happy and tired.

Sentence analogy dataset

This **is** the place to go. $\xrightarrow{\widehat{z_e}}$ This **was** the place to go.

He **comes** home tired and happy. He **came** home happy and tired.

Sentence analogy dataset

$\widehat{z_e}$

This **is** the place to go. → This **was** the place to go.

z_p

He **comes** home tired and happy. He **came** home happy and tired.

Sentence analogy dataset

This **is** the place to go. $\xrightarrow{\widehat{z_e}}$ This **was** the place to go.

z_p

He **comes** home tired and happy.

He **came** home happy and tired.

$\xrightarrow{\widehat{z_e}}$ $\hat{y}$

Sentence analogy dataset

This **is** the place to go. $\xrightarrow{\widehat{z_e}}$ This **was** the place to go.

z_p $\qquad$ y

He **comes** home tired and happy. $\qquad$ He **came** home happy and tired.

$\xrightarrow{\widehat{z_e}}$ $\hat{y}$

Sentence analogy dataset

This **is** the place to go. $\xrightarrow{\widehat{z_e}}$ This **was** the place to go.

z_p

He **comes** home tired and happy.

y

He **came** home happy and tired.

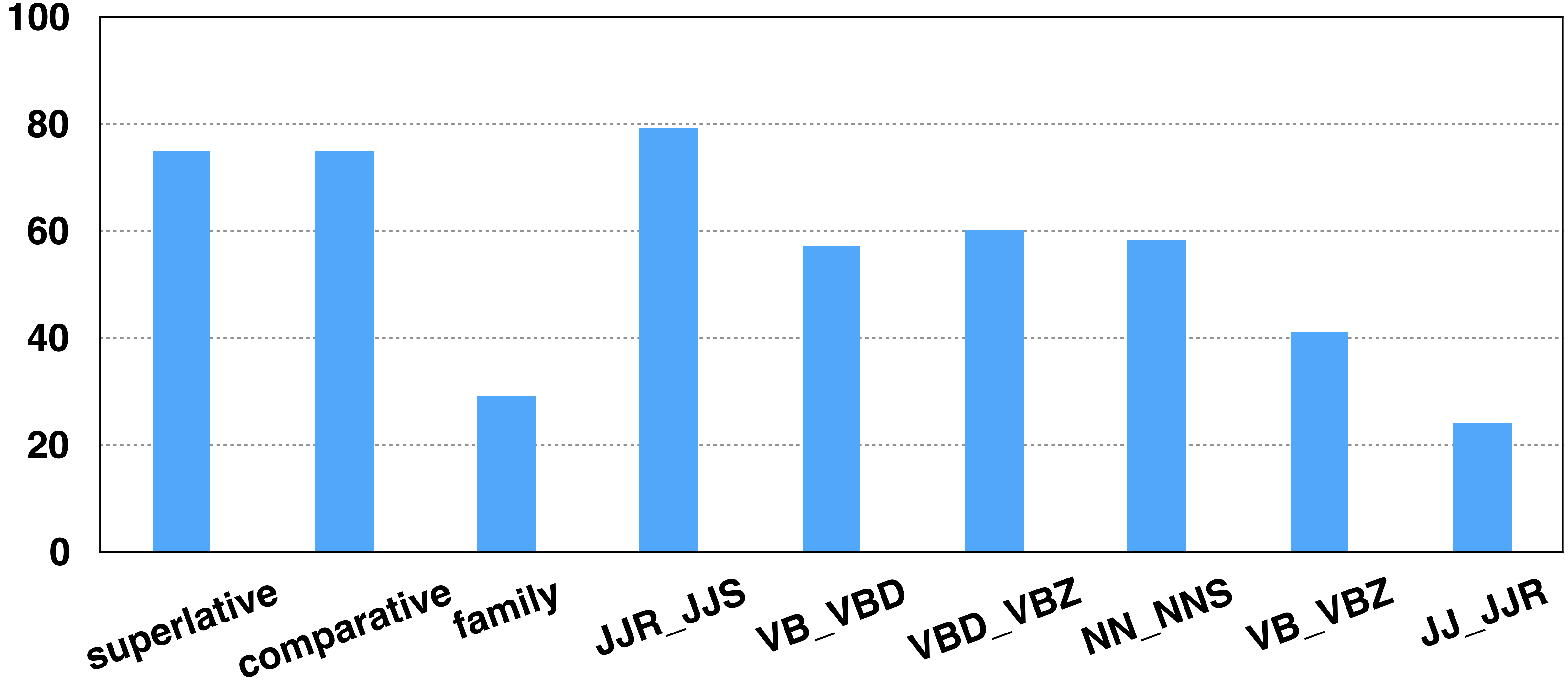
$\updownarrow$ **?**

$\widehat{z_e}$

$\hat{y}$

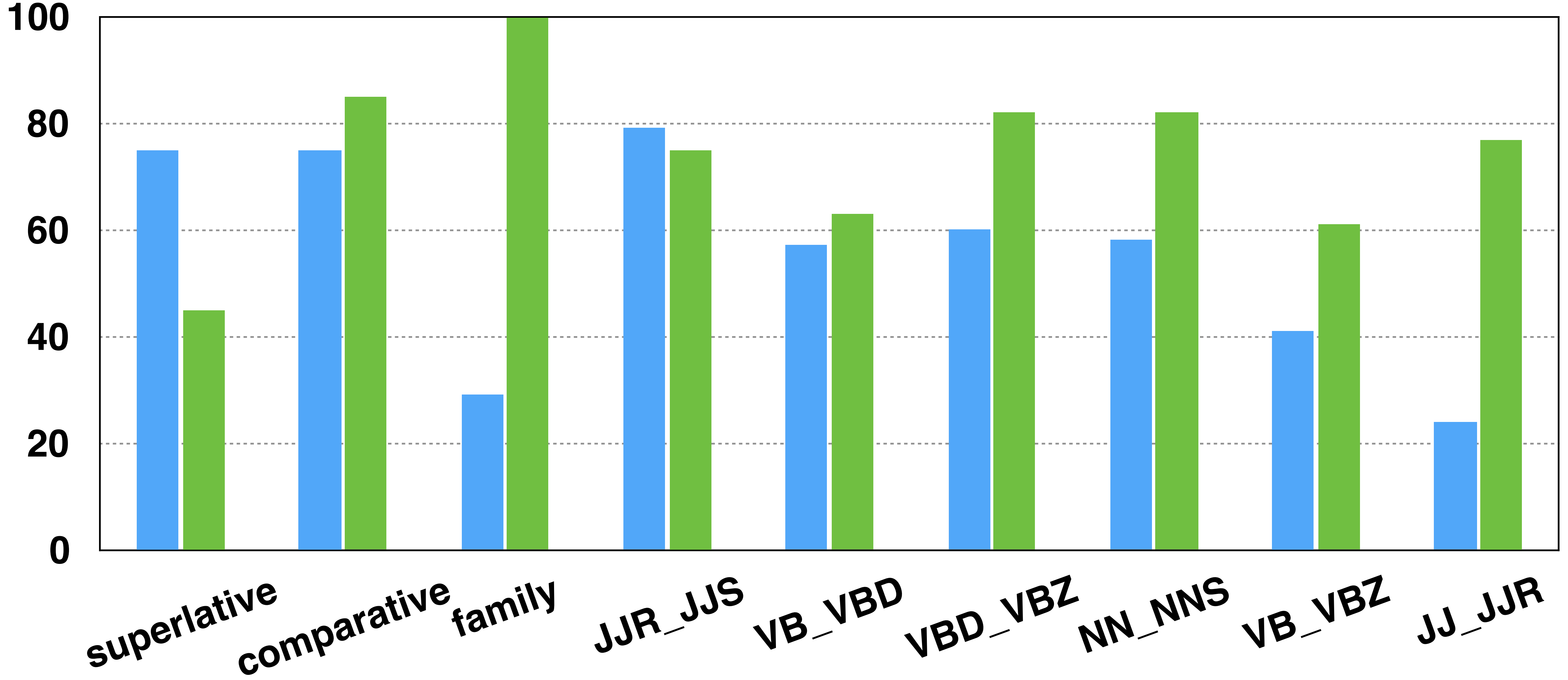
Sentence analogy results

Sentence analogy results



Exact sentence match (top-10 outputs)

Sentence analogy results



blue = exact sentence match (top-10 outputs)

green = exact word match (GloVE)

Results

- **More diverse generations**
- **Higher quality generations**
- **Better perplexity** (BillionWord, Yelp reviews)

✓ **Edits are semantically meaningful**
  - preserve semantic similarity
  - can be used to perform sentence-level analogies

Results

- **More diverse generations**
- **Higher quality generations**

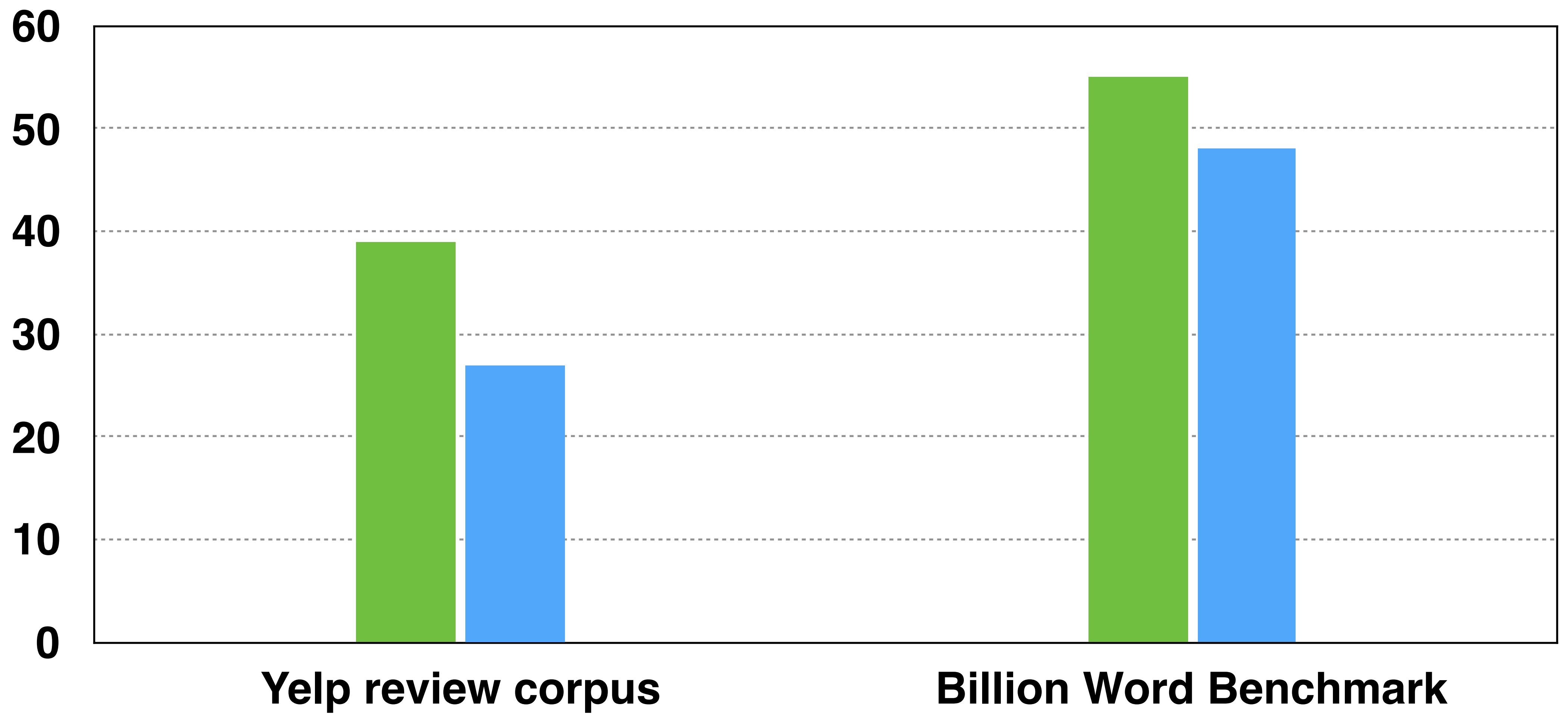
✓ **Better perplexity** (BillionWord, Yelp reviews)

✓ **Edits are semantically meaningful**

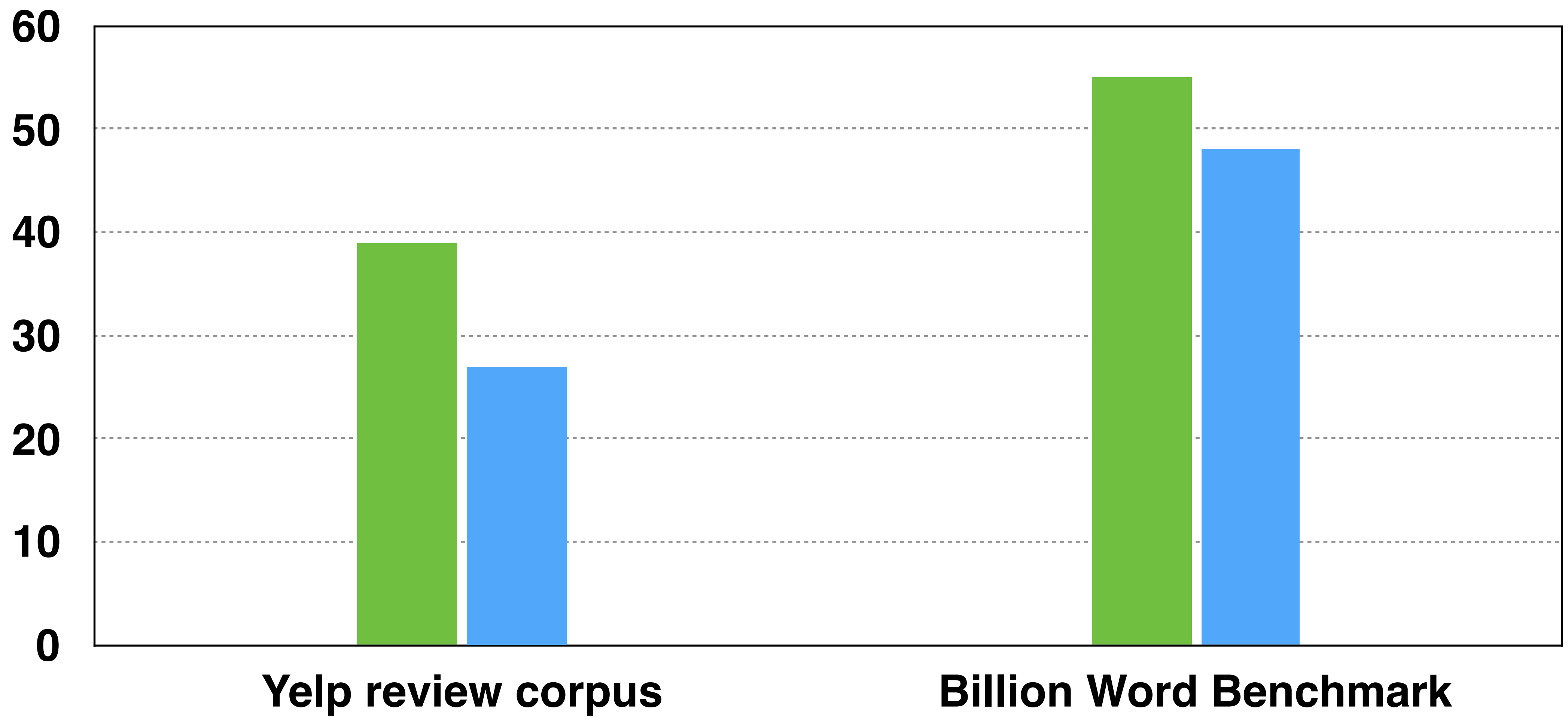
  - preserve semantic similarity
  - can be used to perform sentence-level analogies

Perplexity

Perplexity



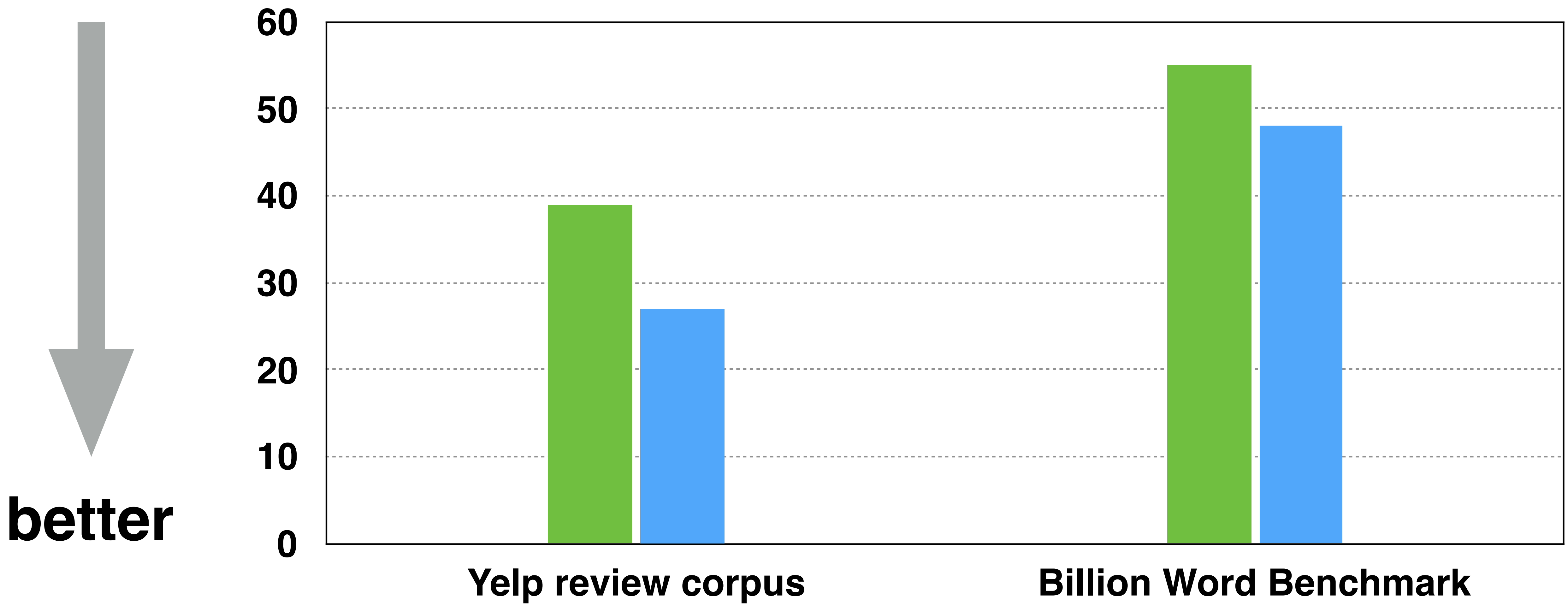
Perplexity



green = standard NLM

blue = NeuralEditor (**same** decoder architecture)
+ backoff to standard NLM

Perplexity

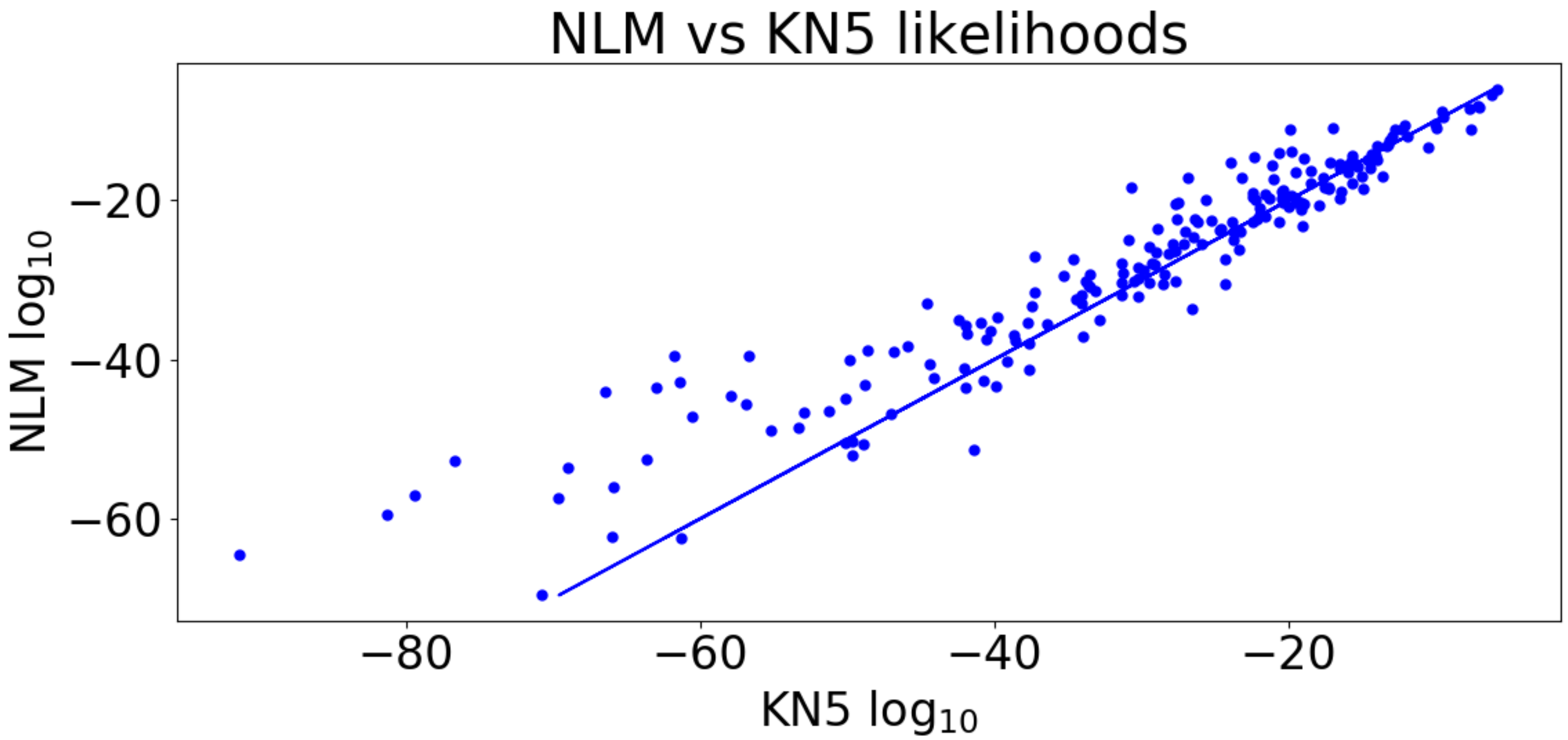


green = standard NLM

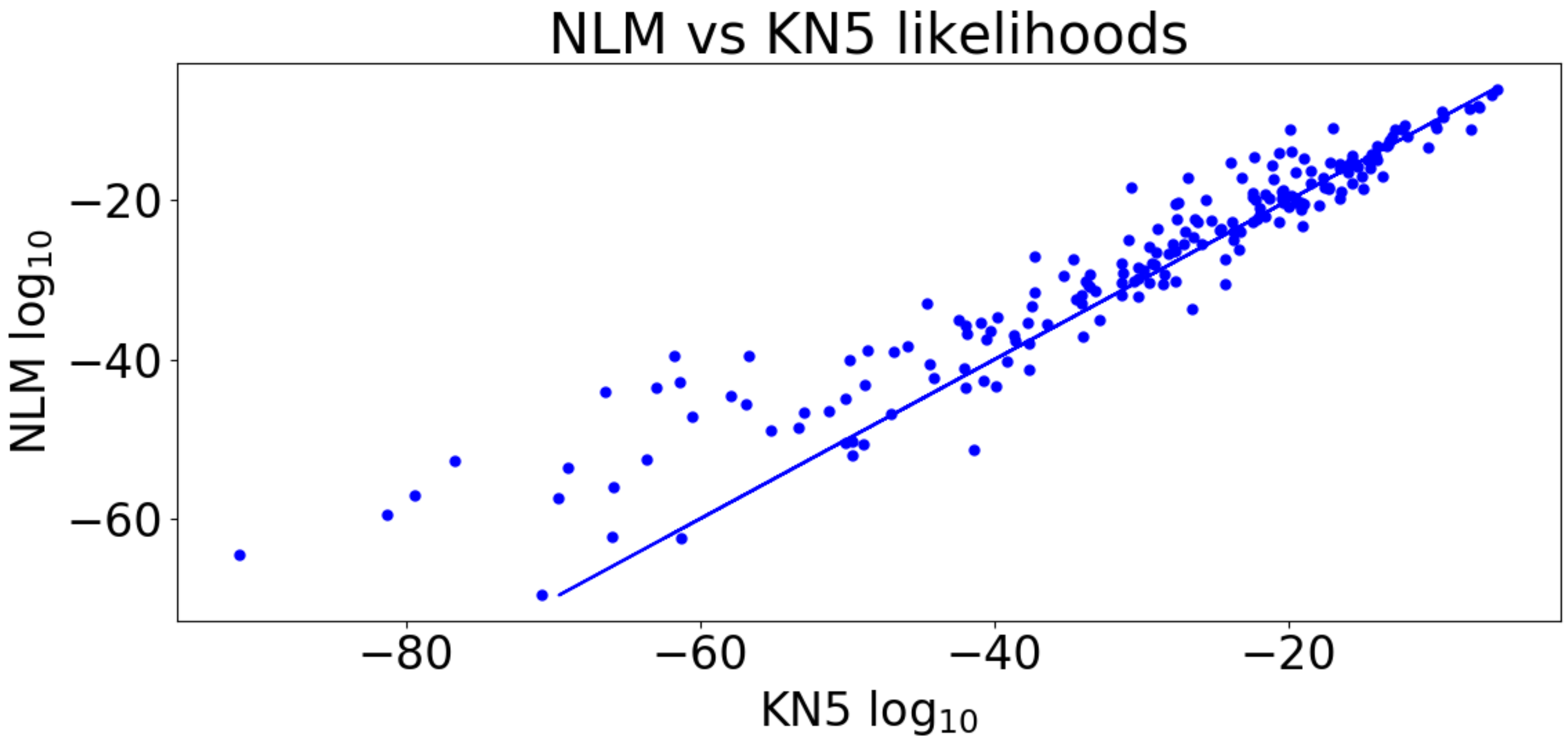
blue = NeuralEditor (**same** decoder architecture)
+ backoff to standard NLM

Perplexity (closer look)

Perplexity (closer look)

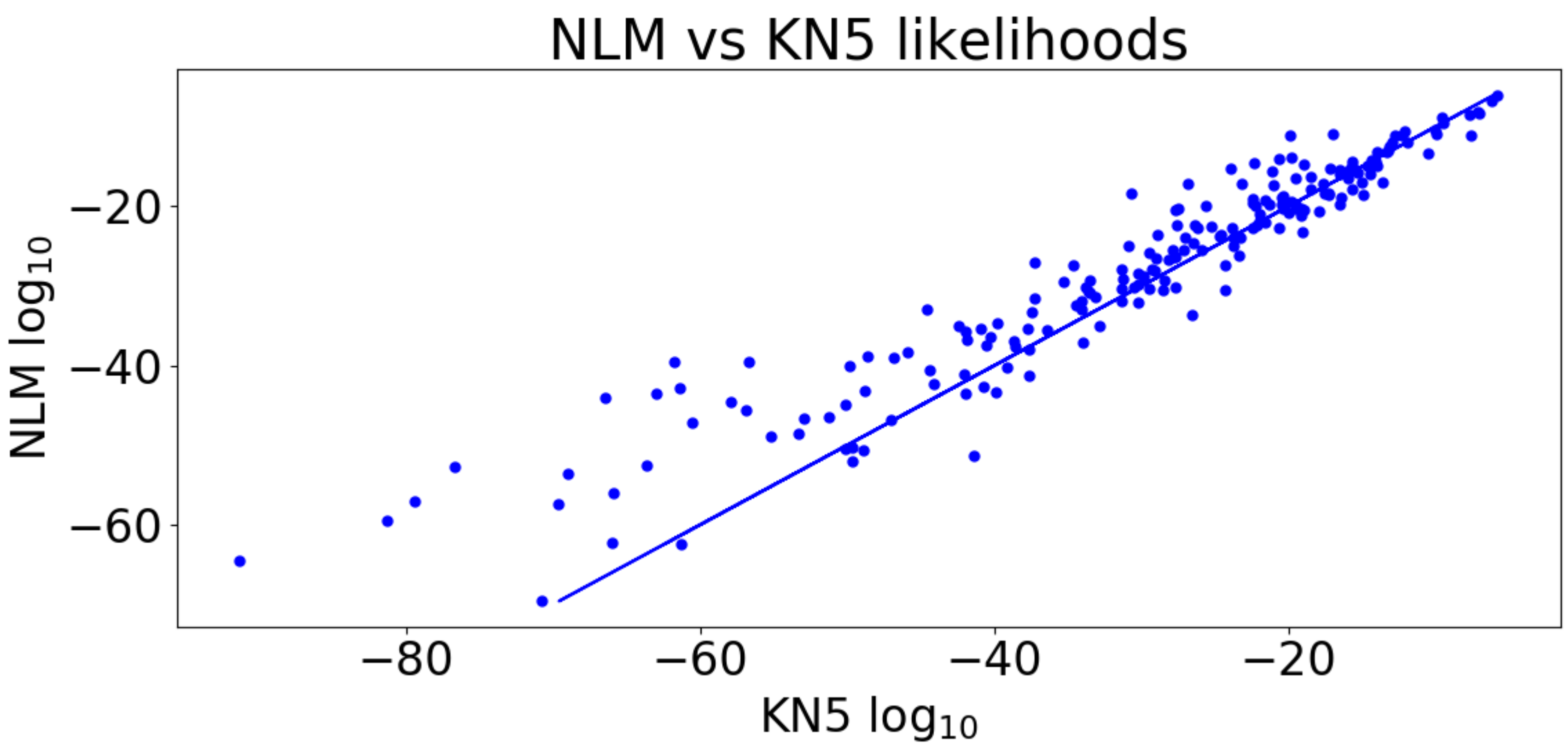


Perplexity (closer look)



neural LM

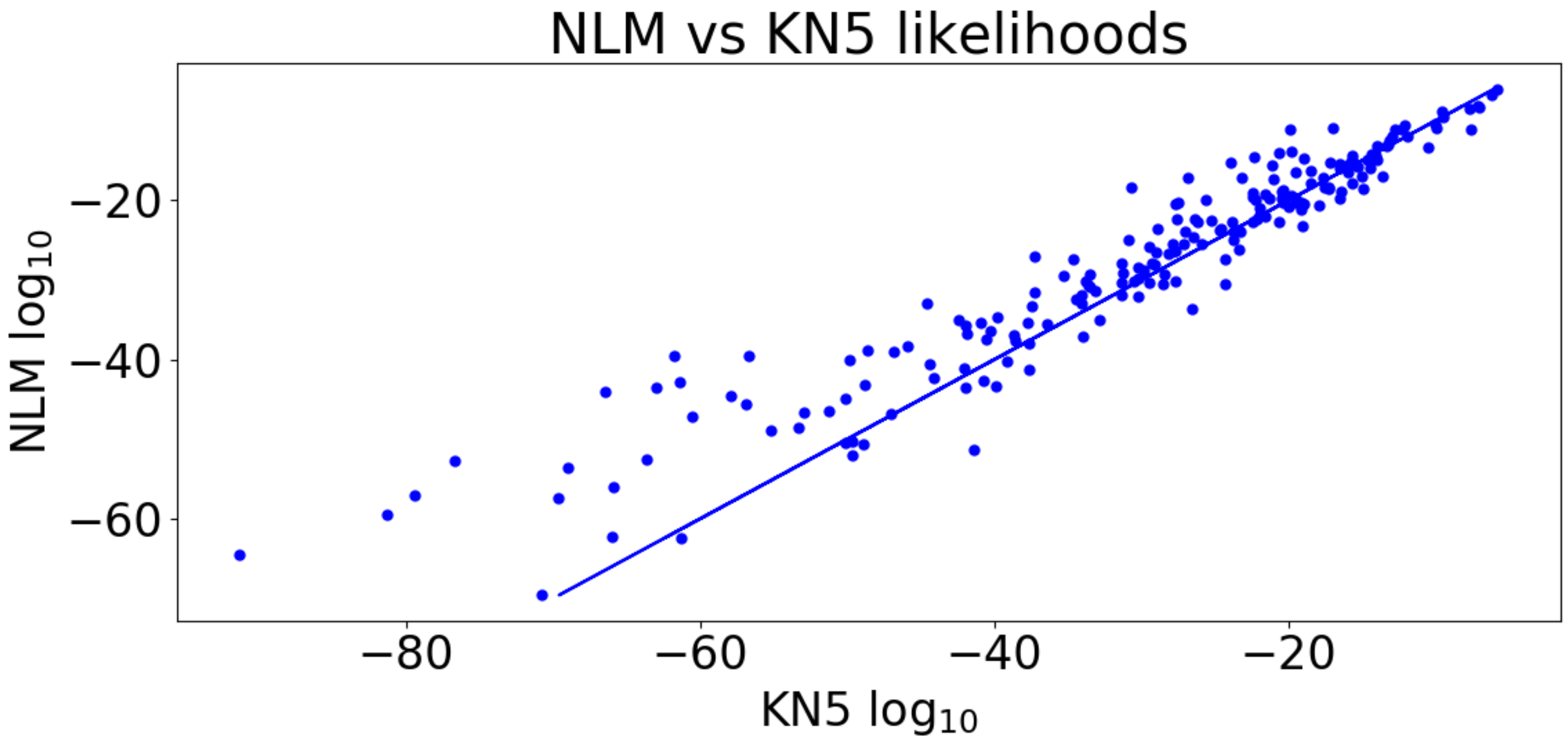
Perplexity (closer look)



neural LM

classic Kneser-Ney LM

Perplexity (closer look)

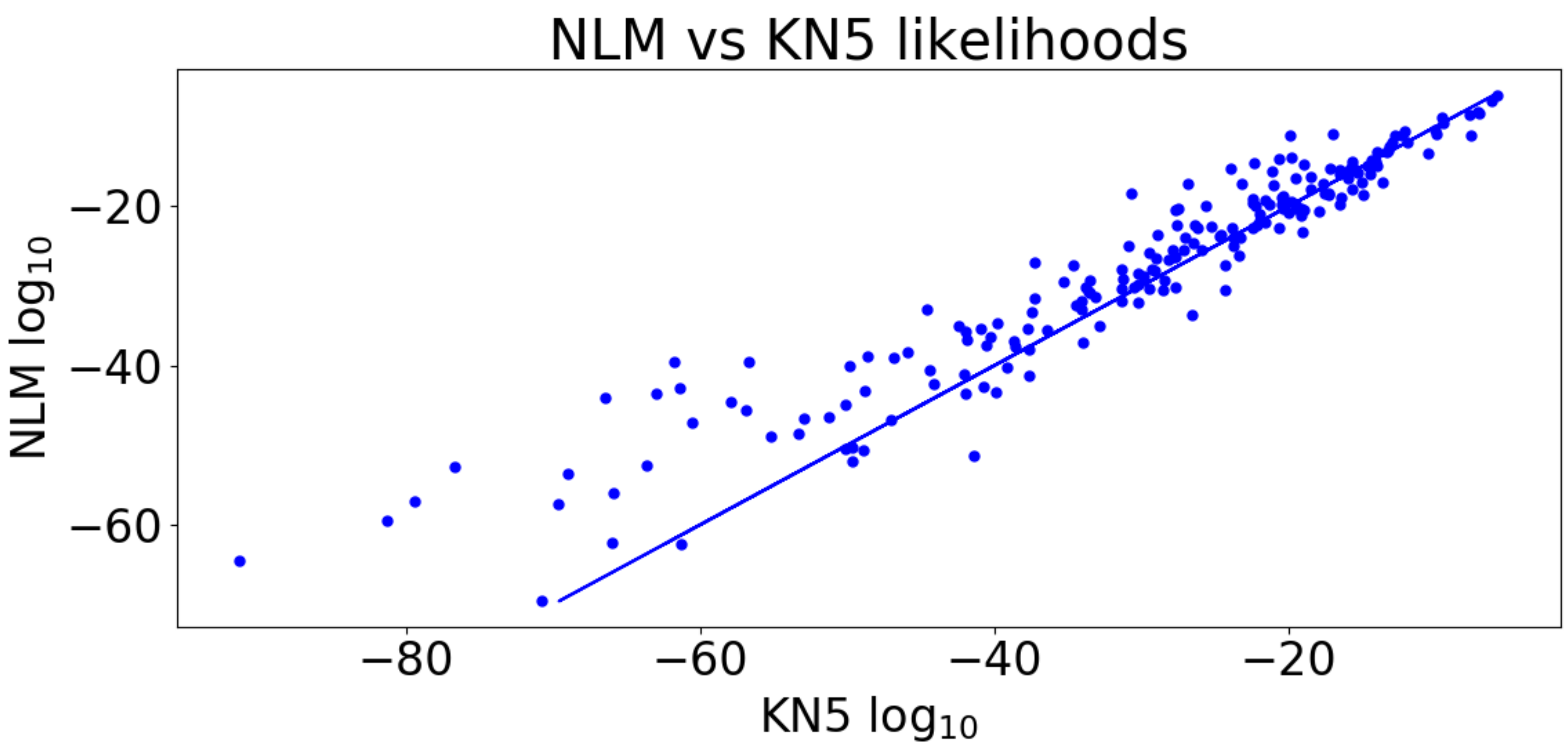


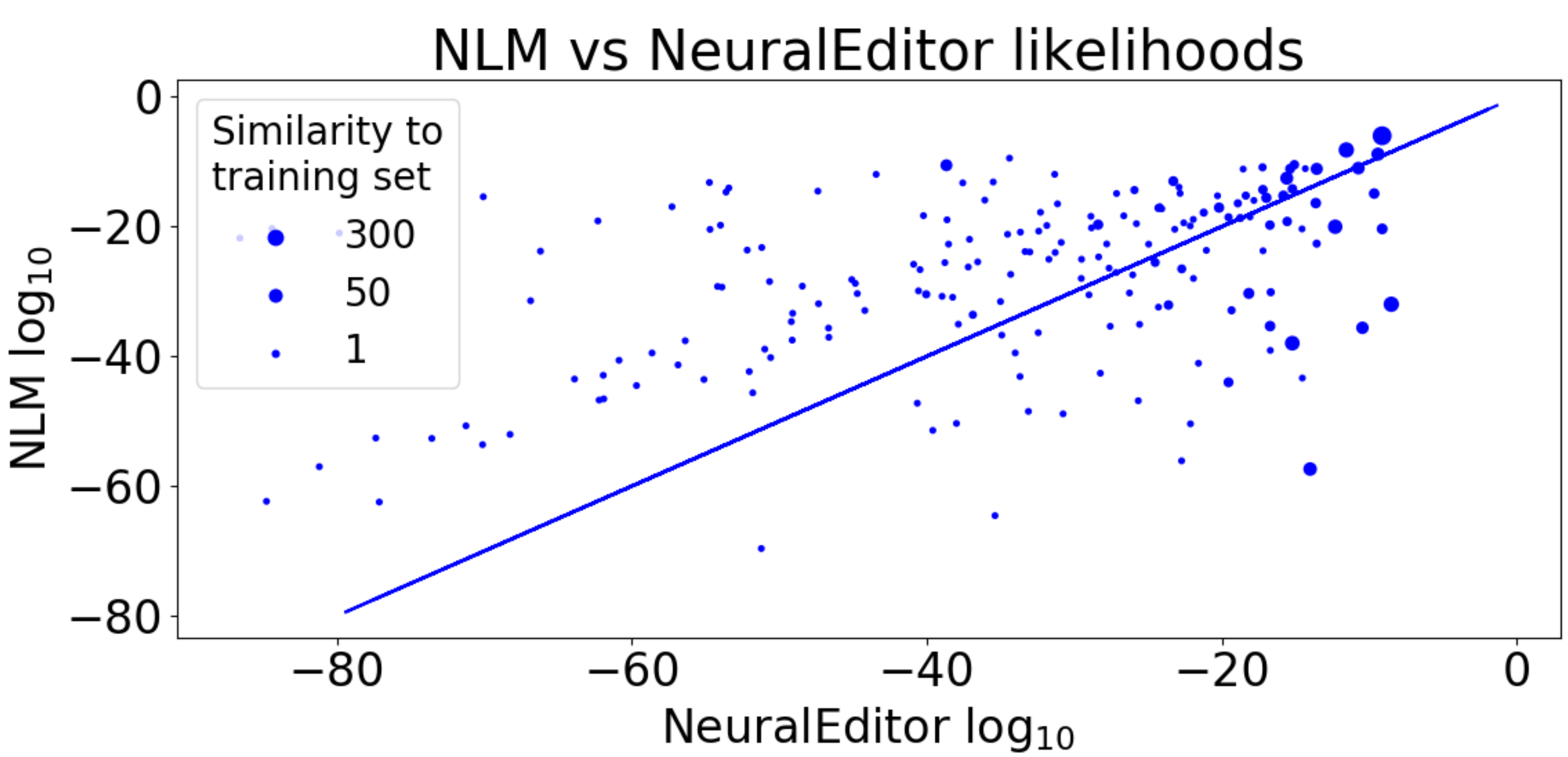
neural LM

classic Kneser-Ney LM

similar

Perplexity (closer look)



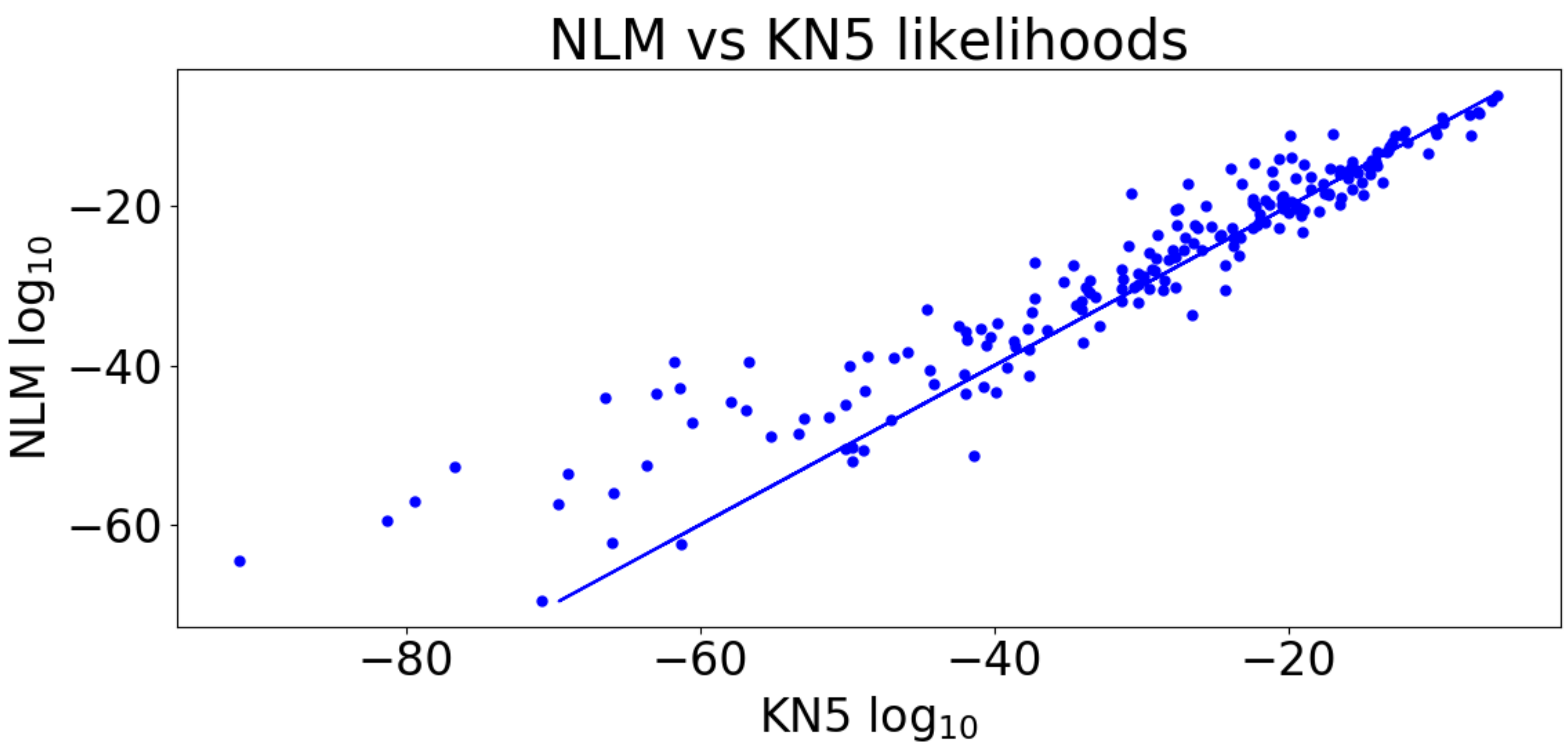


neural LM

classic Kneser-Ney LM

similar

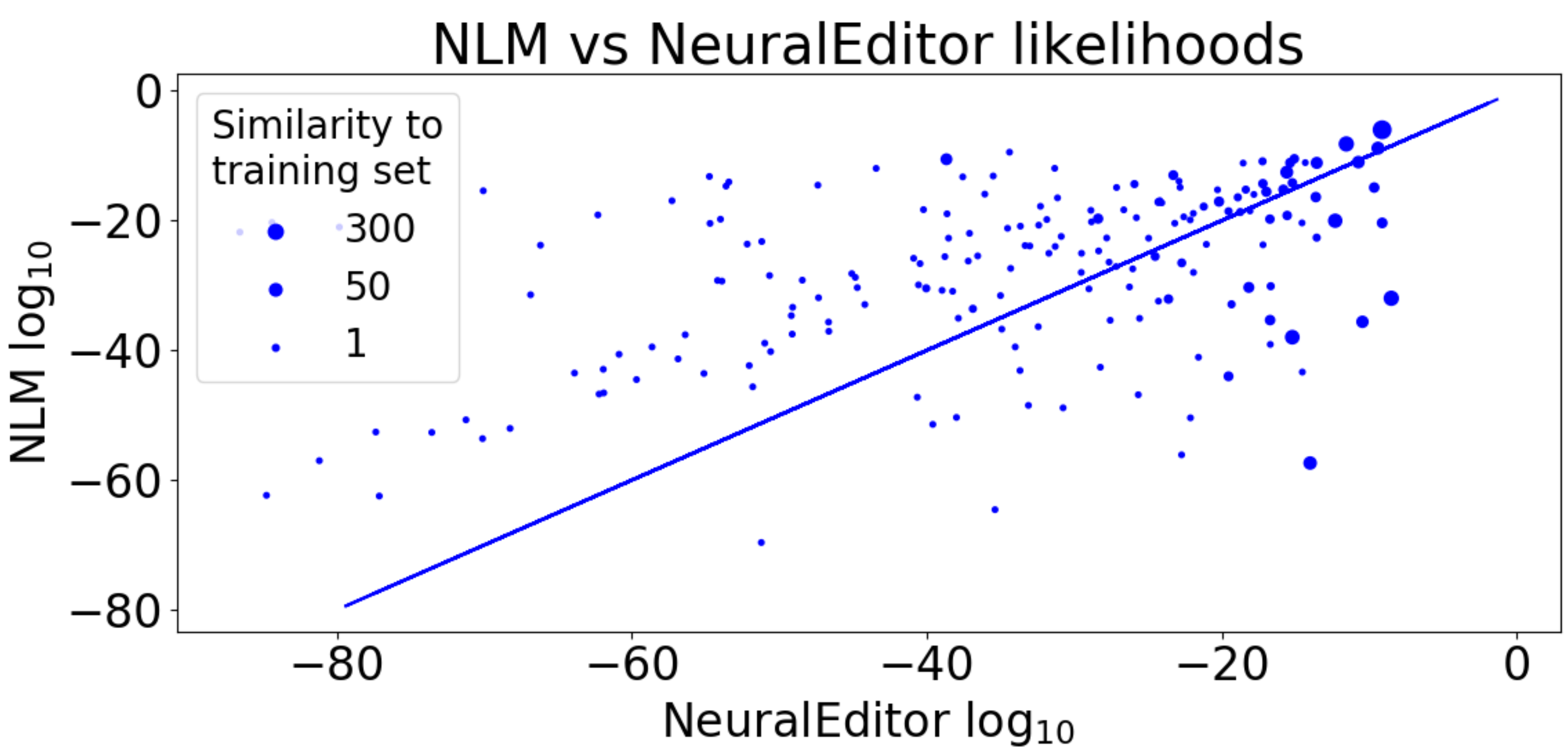
Perplexity (closer look)



neural LM

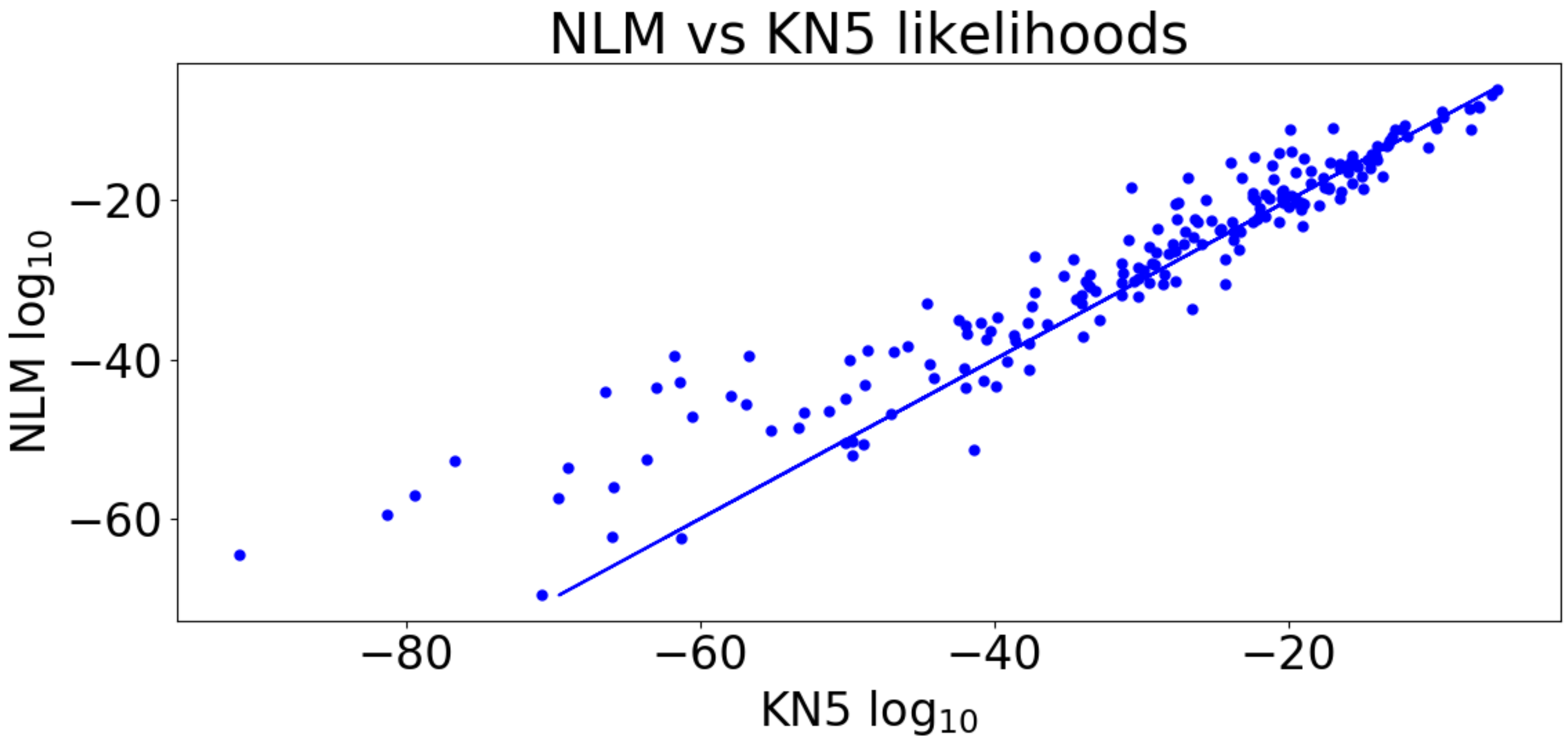
classic Kneser-Ney LM

similar



neural LM

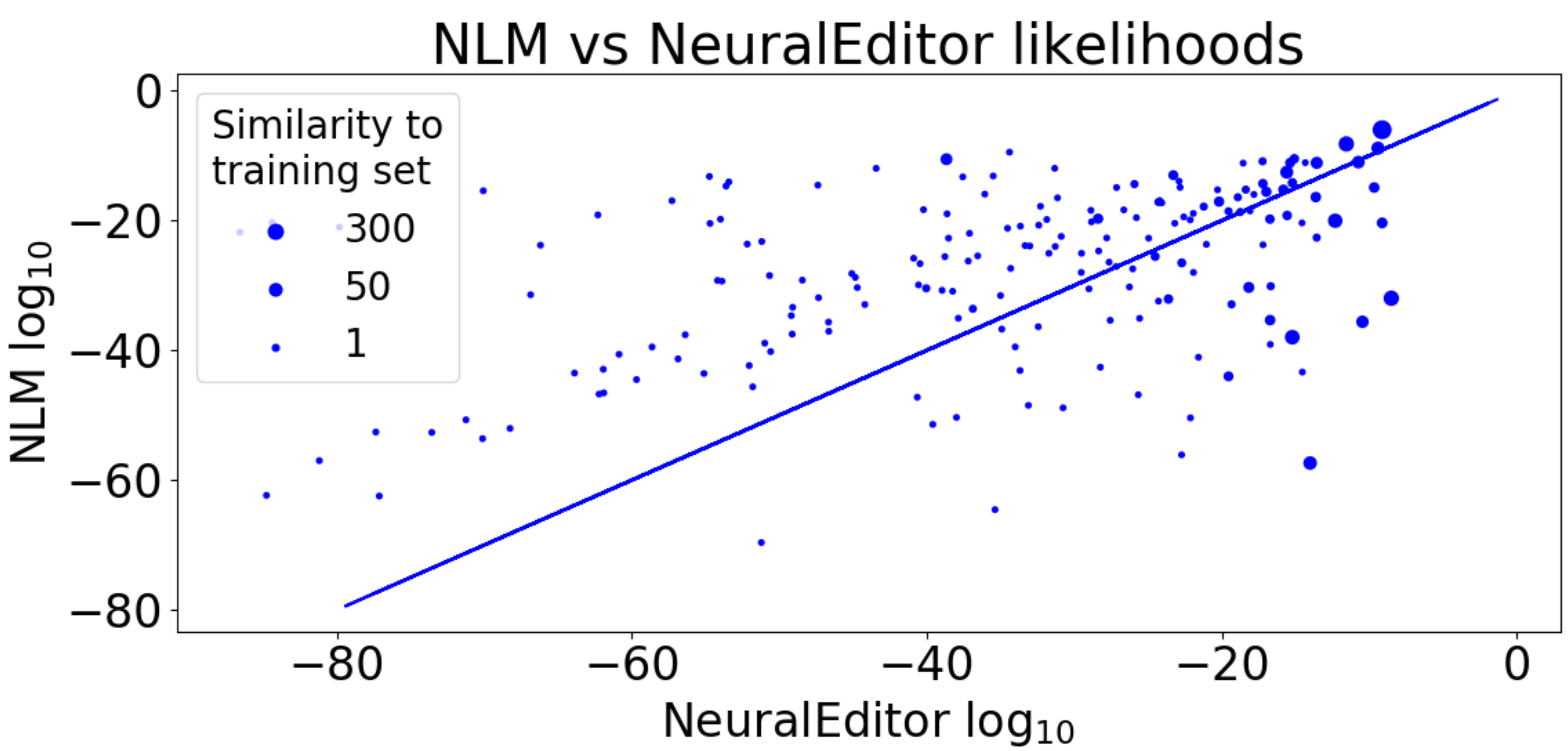
Perplexity (closer look)



neural LM

classic Kneser-Ney LM

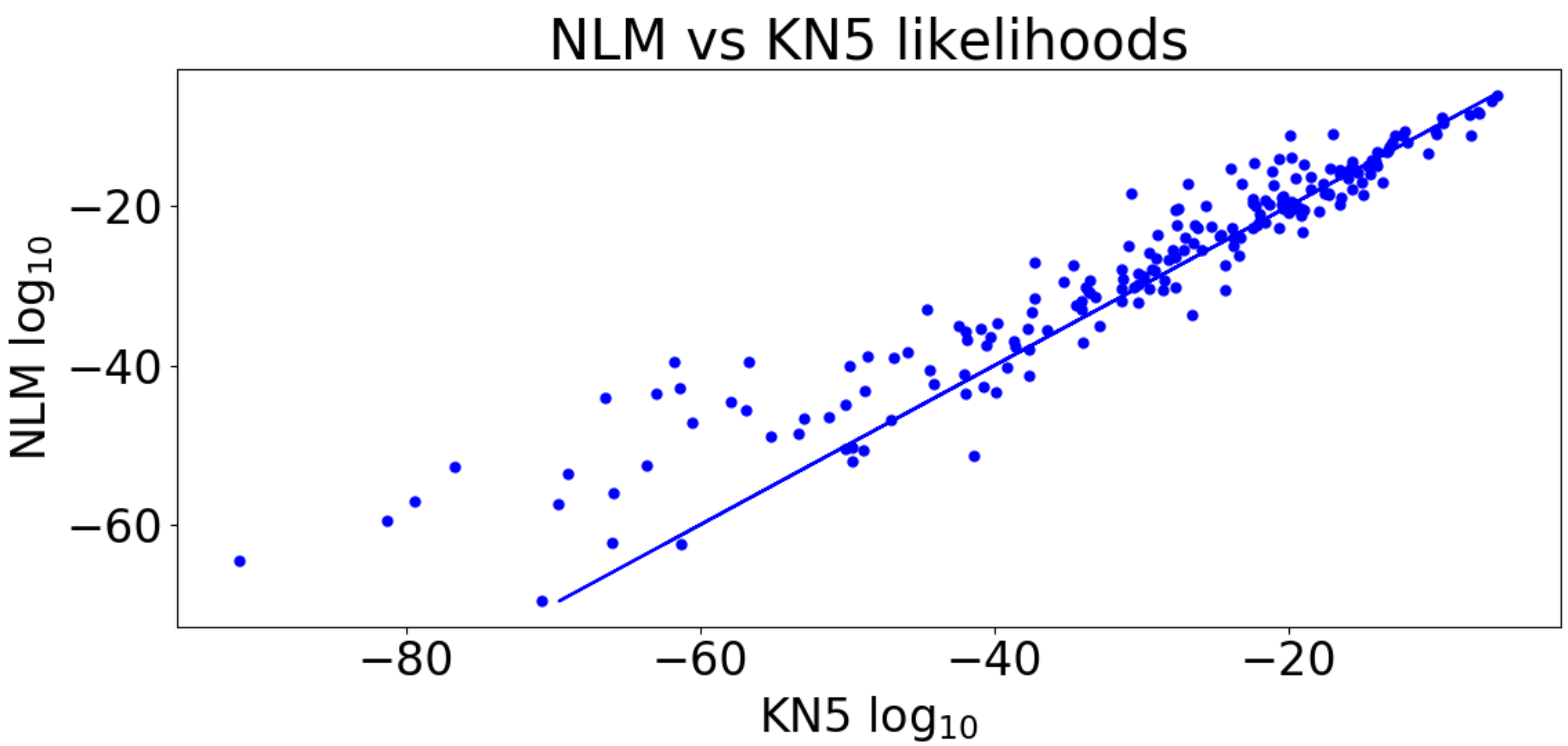
similar



neural LM

NeuralEditor

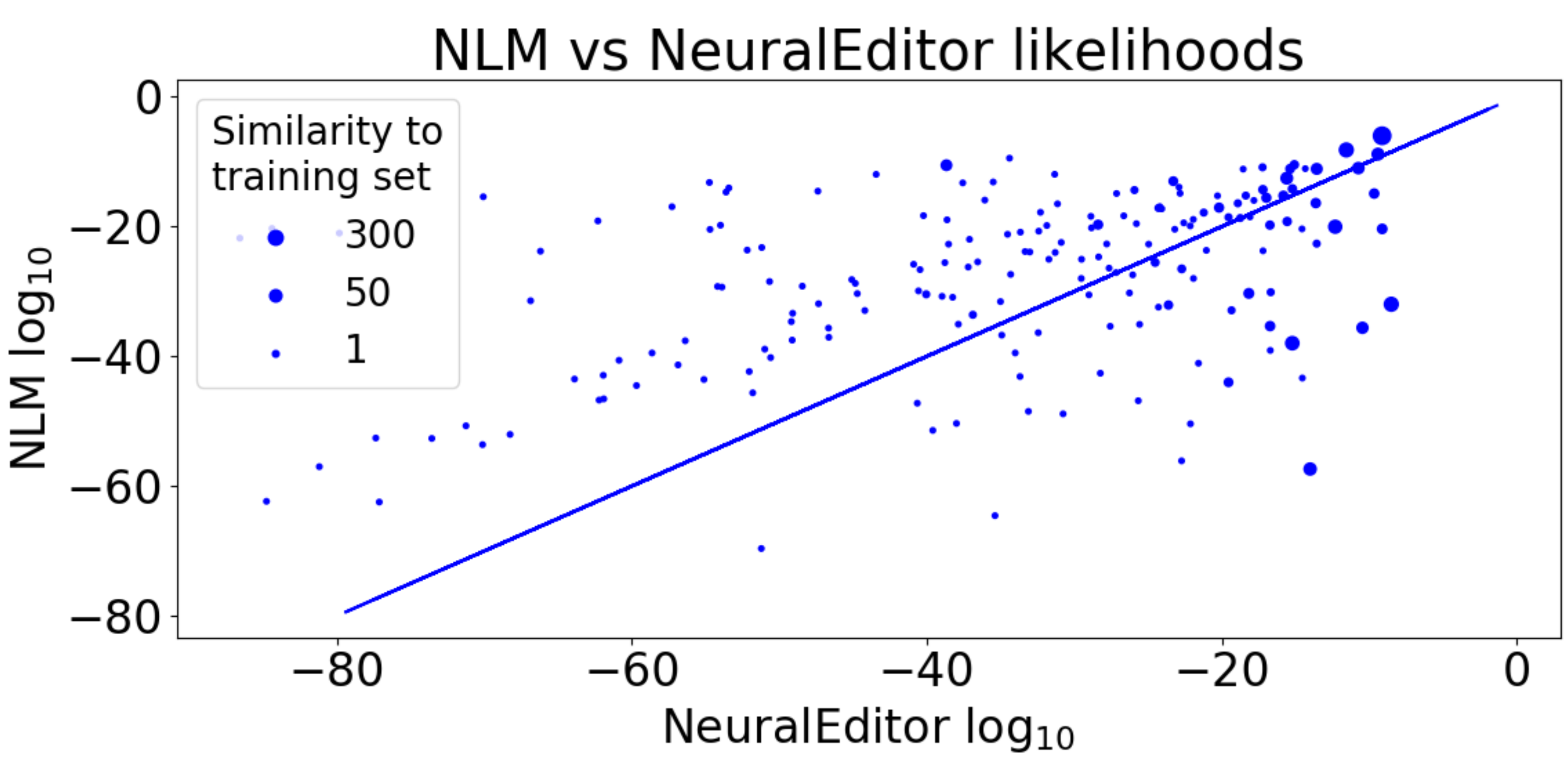
Perplexity (closer look)



neural LM

classic Kneser-Ney LM

similar



neural LM

NeuralEditor

different

Results

- **More diverse generations**
- **Higher quality generations**

✓ **Better perplexity** (BillionWord, Yelp reviews)

✓ **Edits are semantically meaningful**

- preserve semantic similarity
- can be used to perform sentence-level analogies

Results

✓ **More diverse generations**

✓ **Higher quality generations**

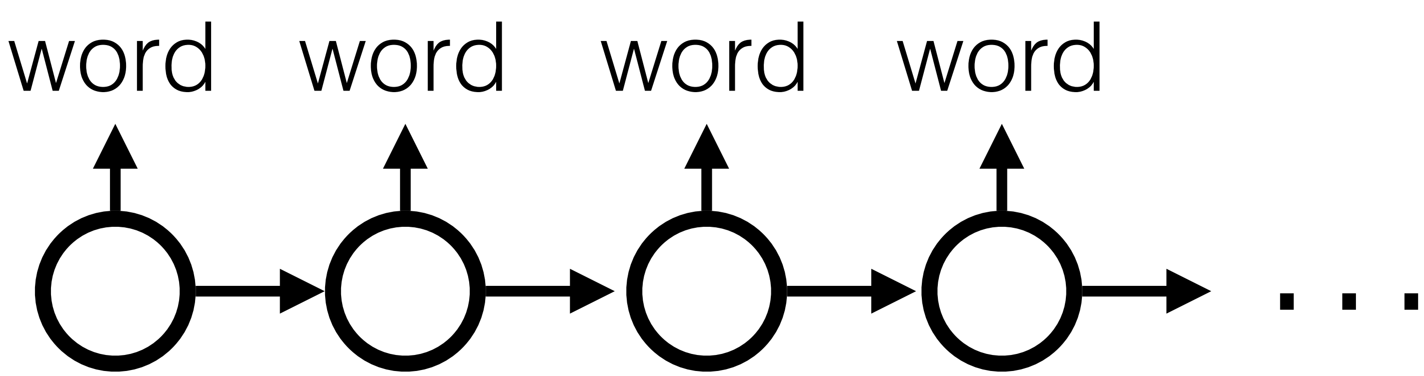
✓ **Better perplexity** (BillionWord, Yelp reviews)

✓ **Edits are semantically meaningful**

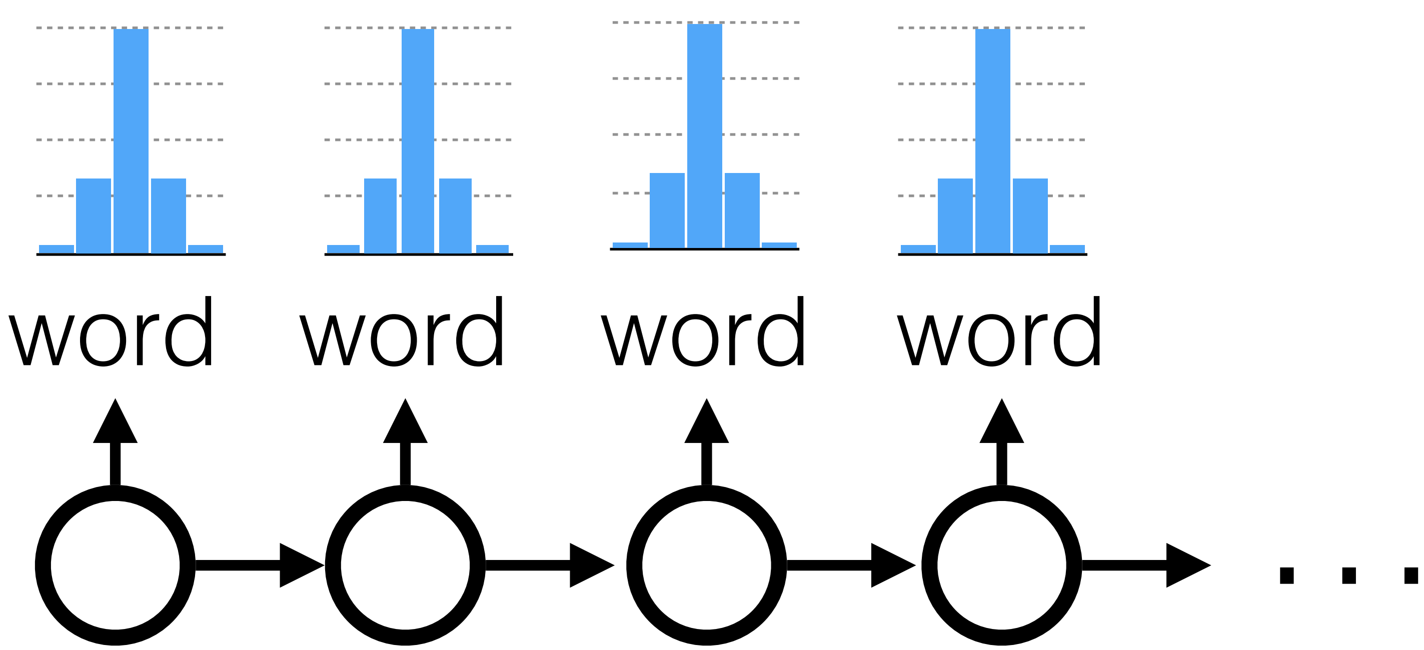
- preserve semantic similarity
- can be used to perform sentence-level analogies

Naive way to increase diversity

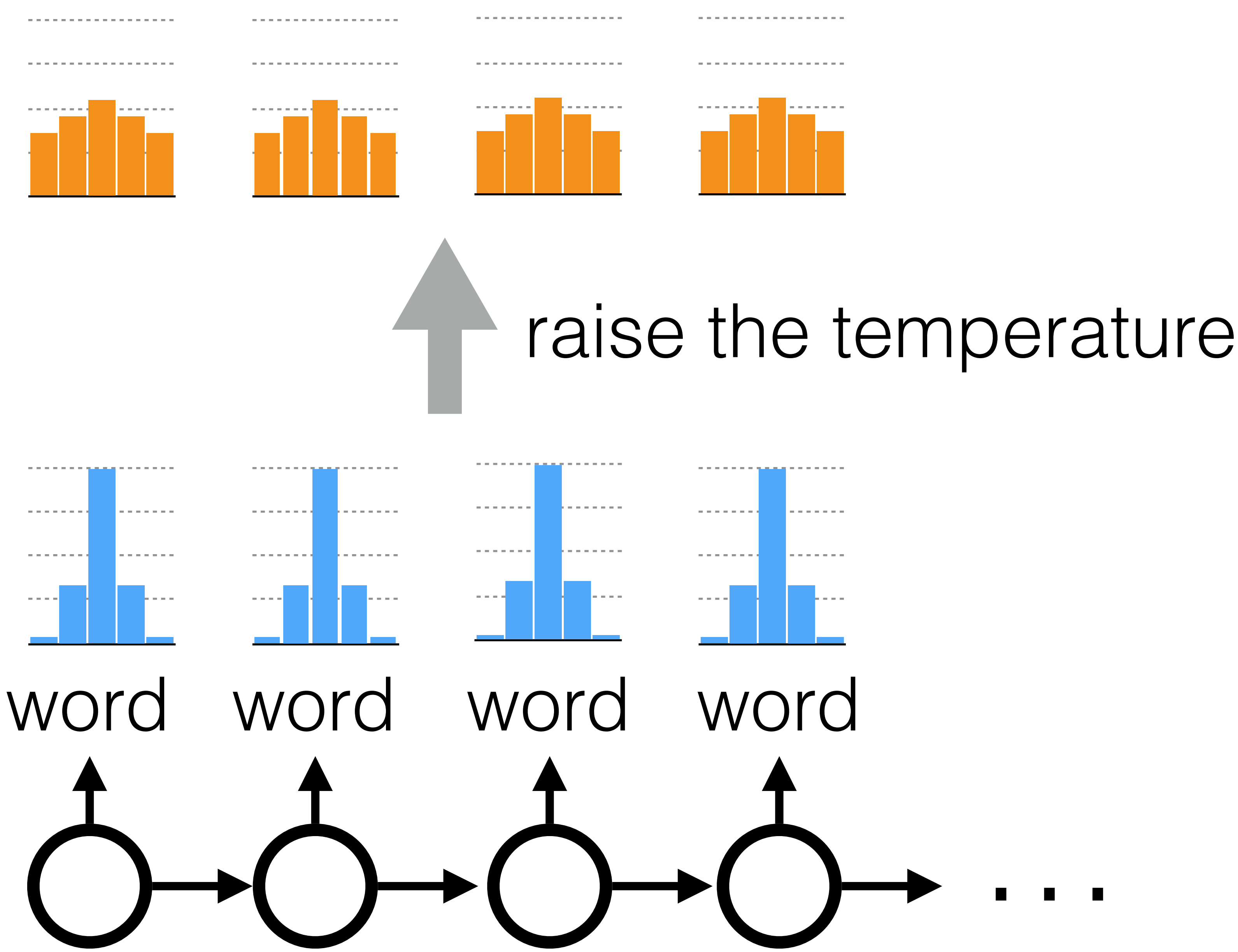
Naive way to increase diversity



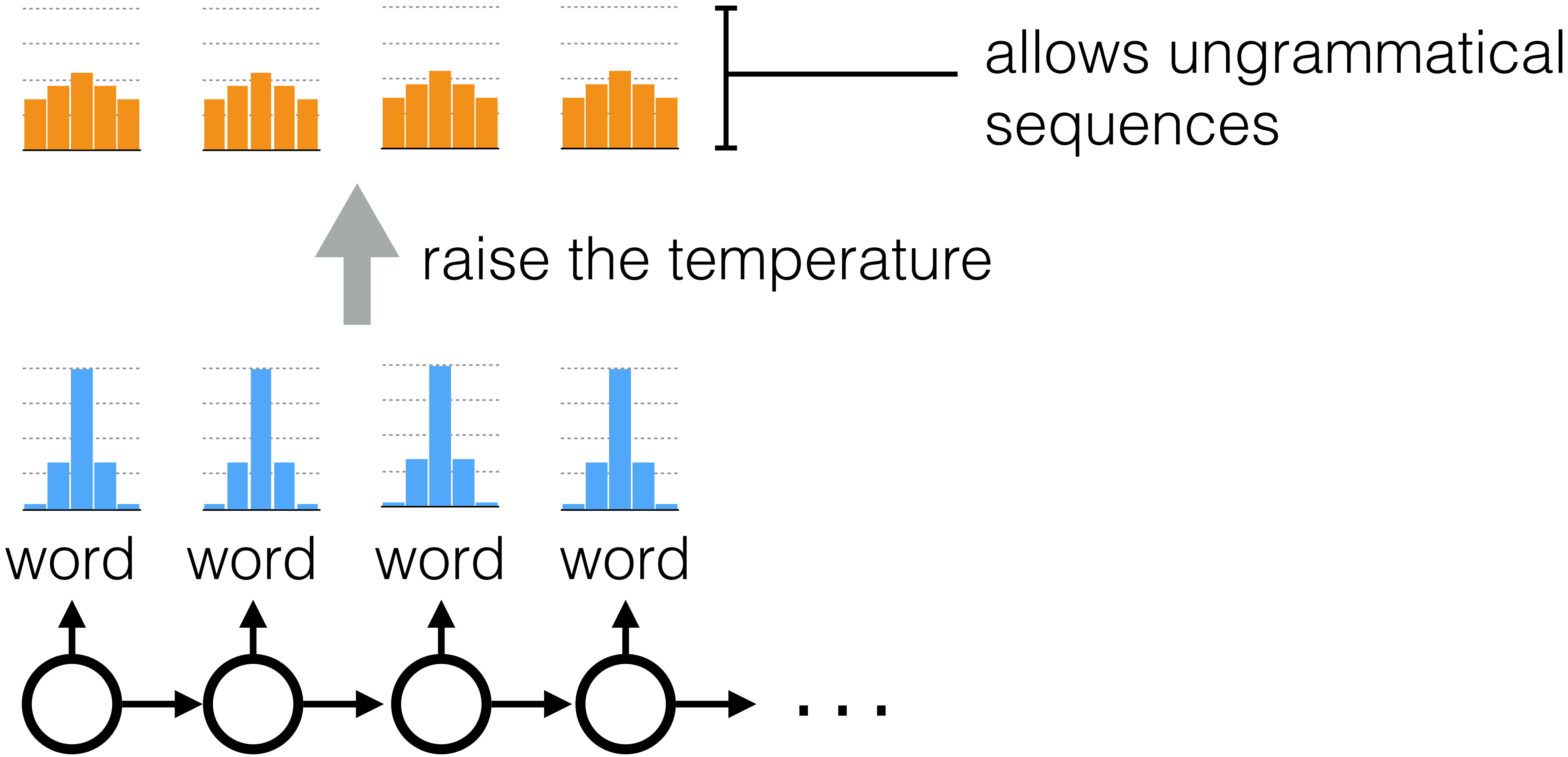
Naive way to increase diversity



Naive way to increase diversity



Naive way to increase diversity



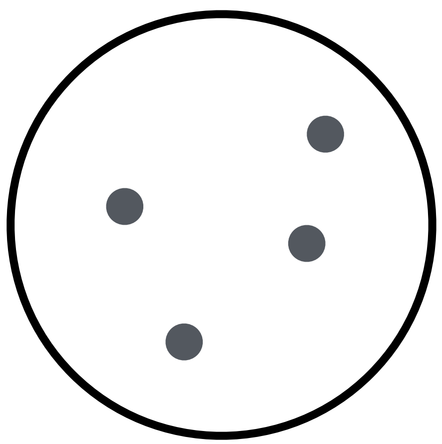
Increasing diversity of NeuralEditor

Increasing diversity of NeuralEditor

$$z_p \sim p_{\text{proto}}$$

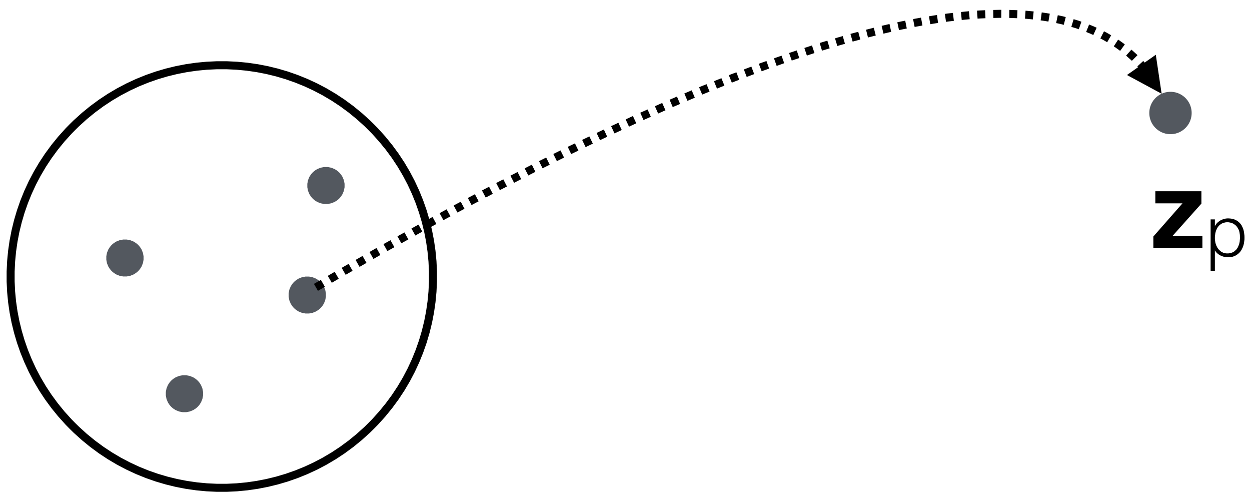
Increasing diversity of NeuralEditor

$$z_p \sim p_{\text{proto}}$$

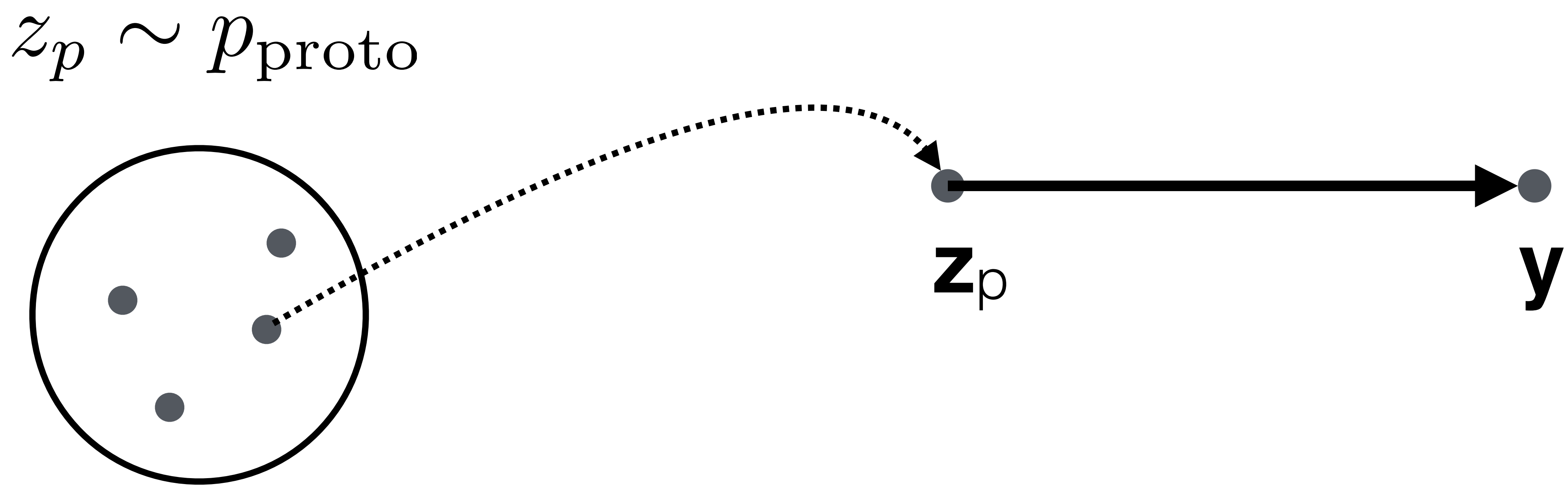


Increasing diversity of NeuralEditor

$z_p \sim p_{\text{proto}}$

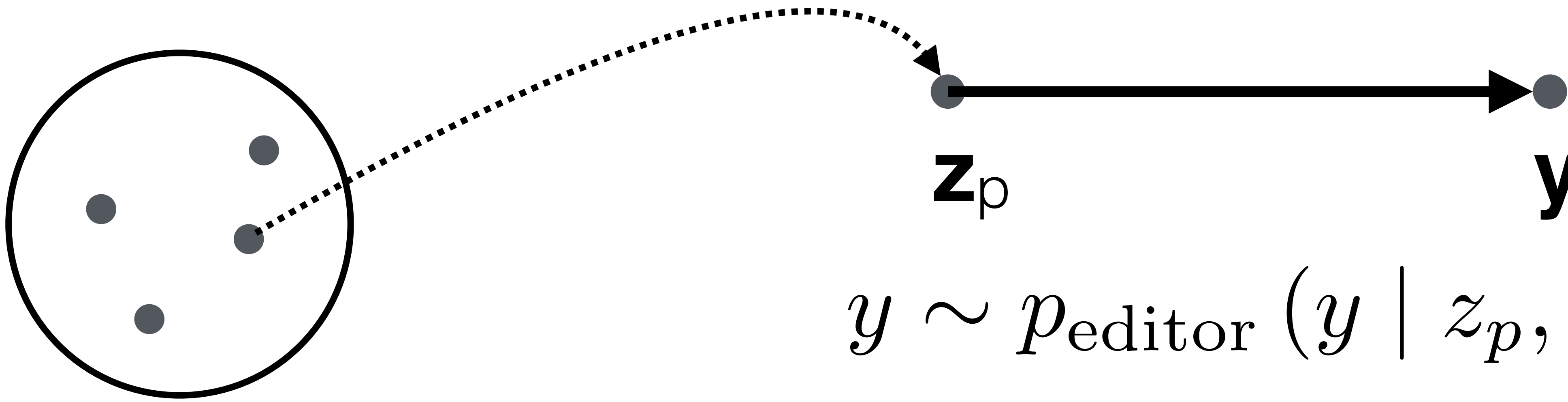


Increasing diversity of NeuralEditor



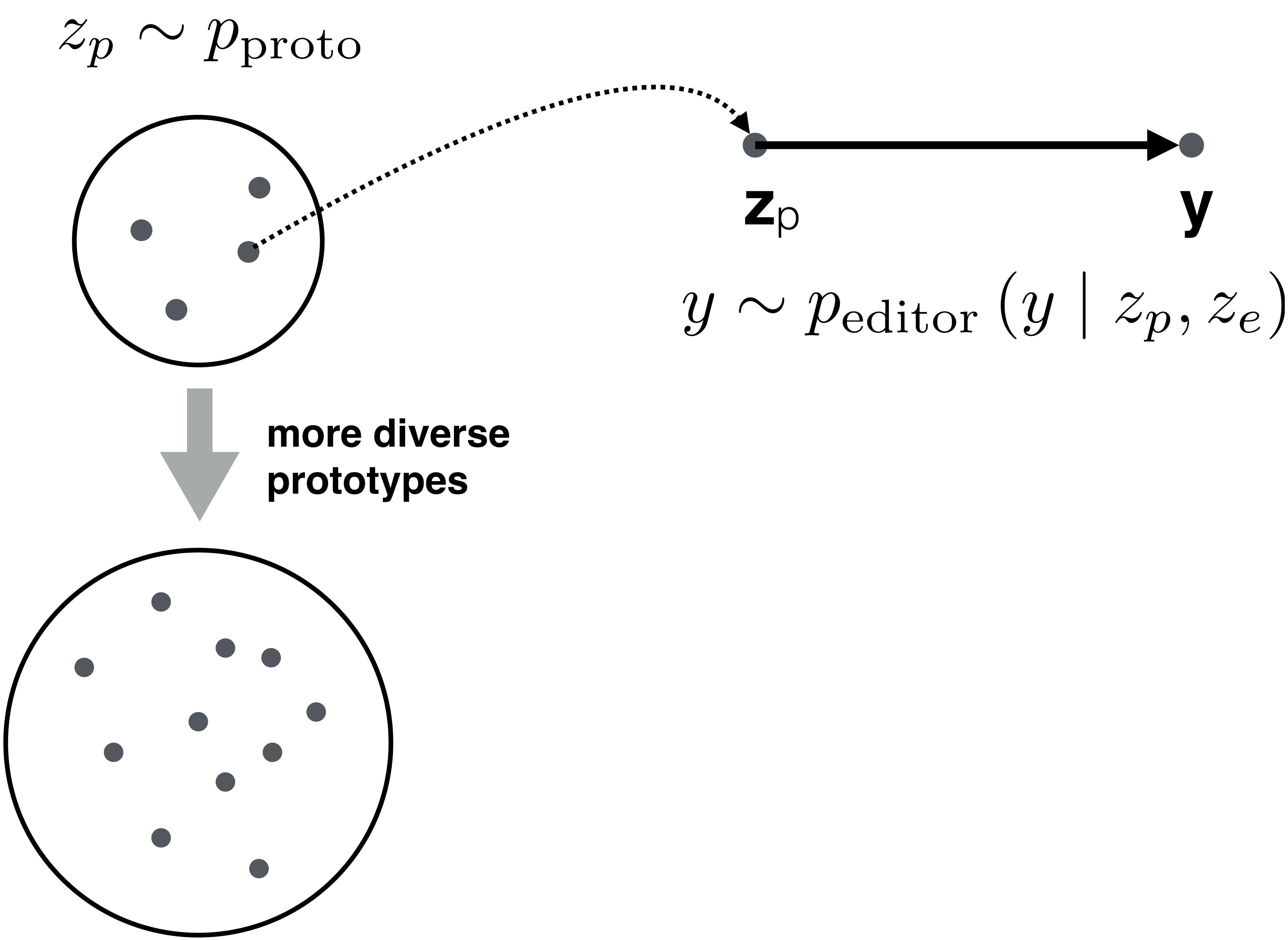
Increasing diversity of NeuralEditor

$$z_p \sim p_{\text{proto}}$$

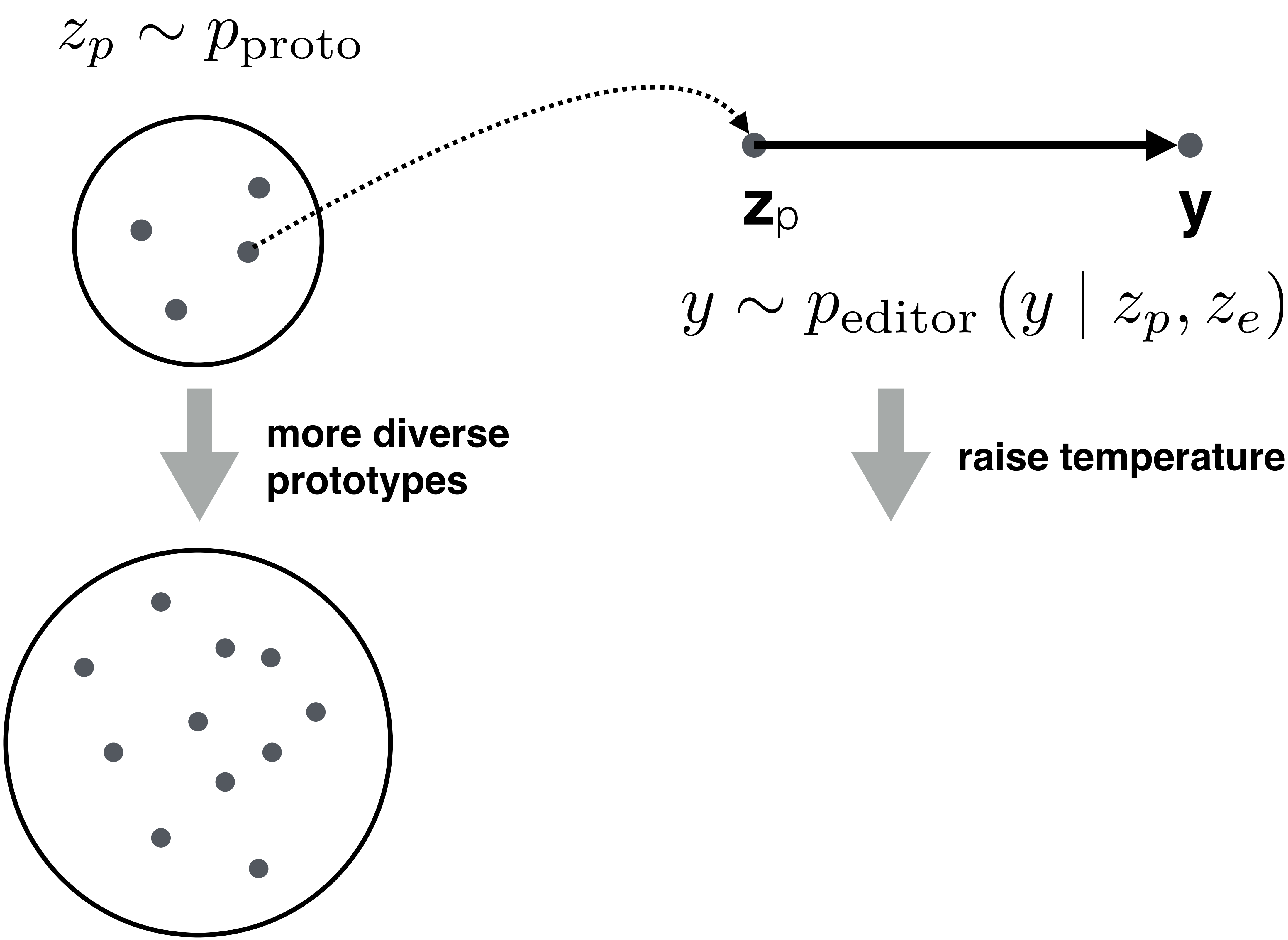


$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

Increasing diversity of NeuralEditor

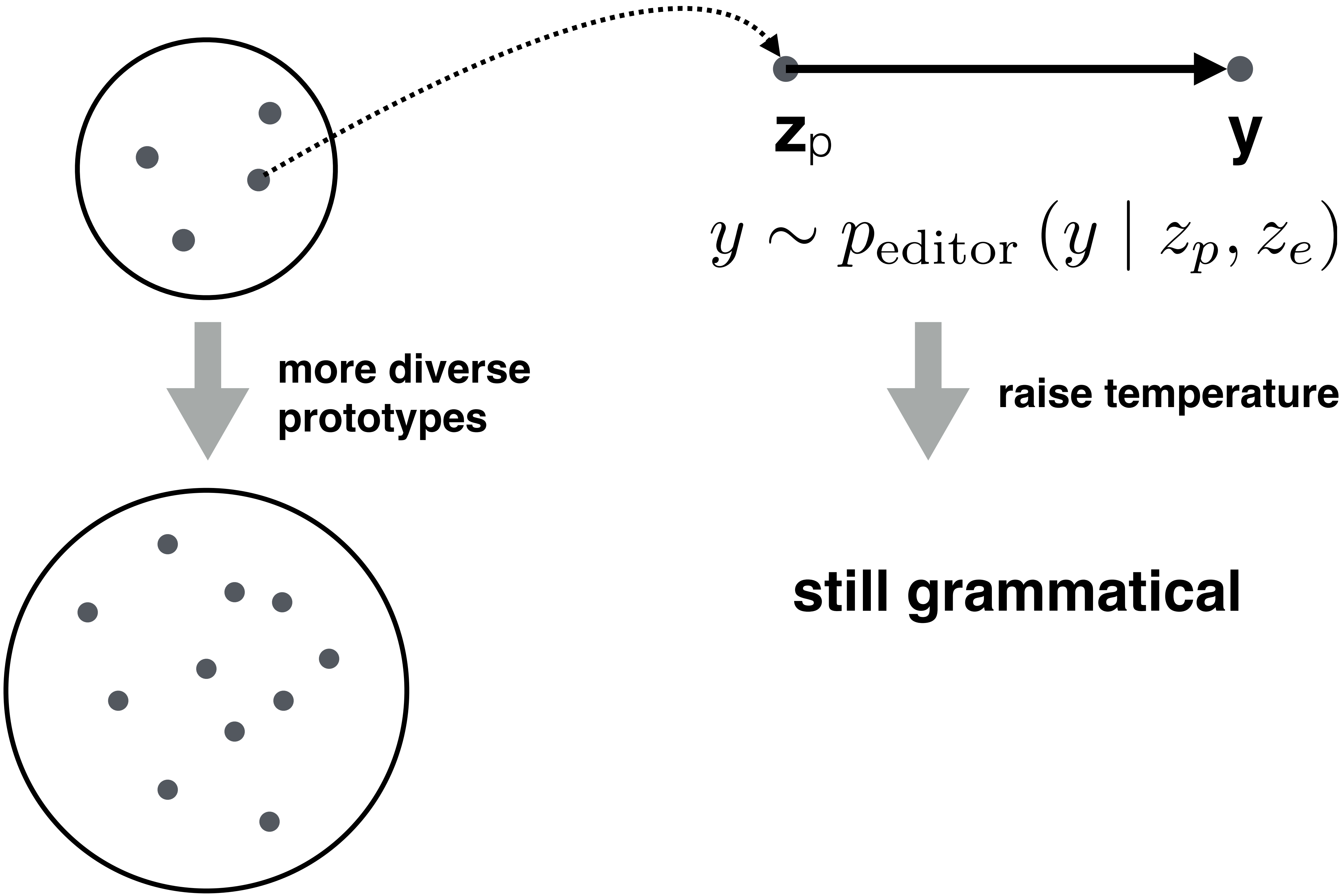


Increasing diversity of NeuralEditor



Increasing diversity of NeuralEditor

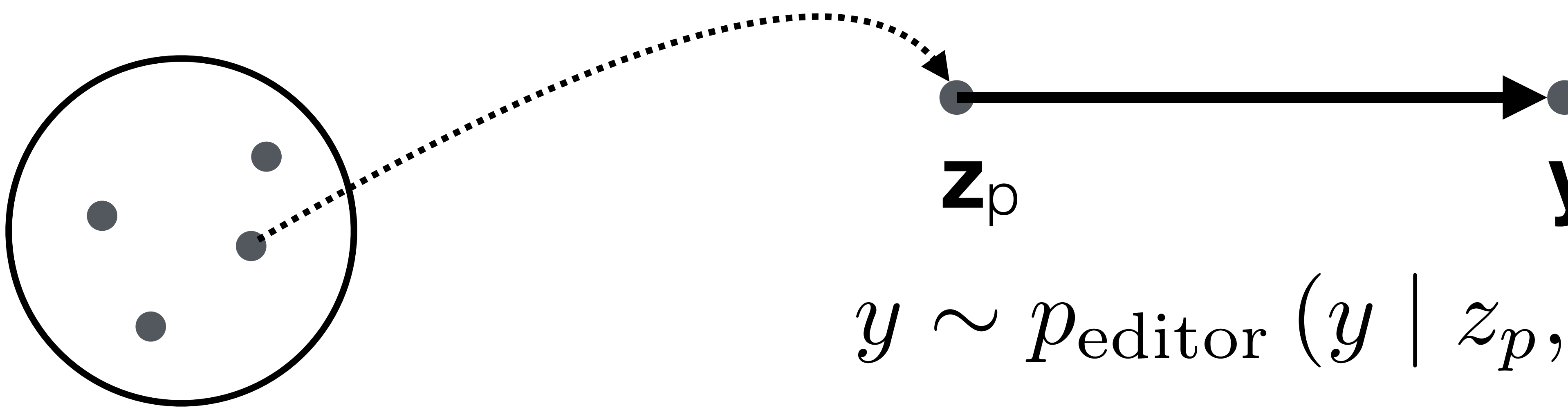
$$z_p \sim p_{\mathrm{proto}}$$



$$y \sim p_{\mathrm{editor}}\left(y \mid z_p, z_e\right)$$

Increasing diversity of NeuralEditor

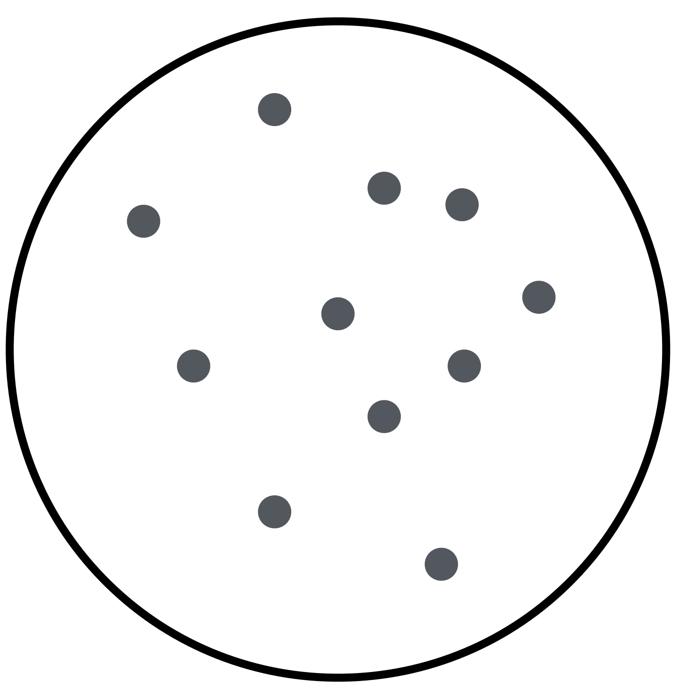
$z_p \sim p_{\text{proto}}$



$$y \sim p_{\text{editor}}\left(y \mid z_p, z_e\right)$$

more diverse prototypes

raise temperature

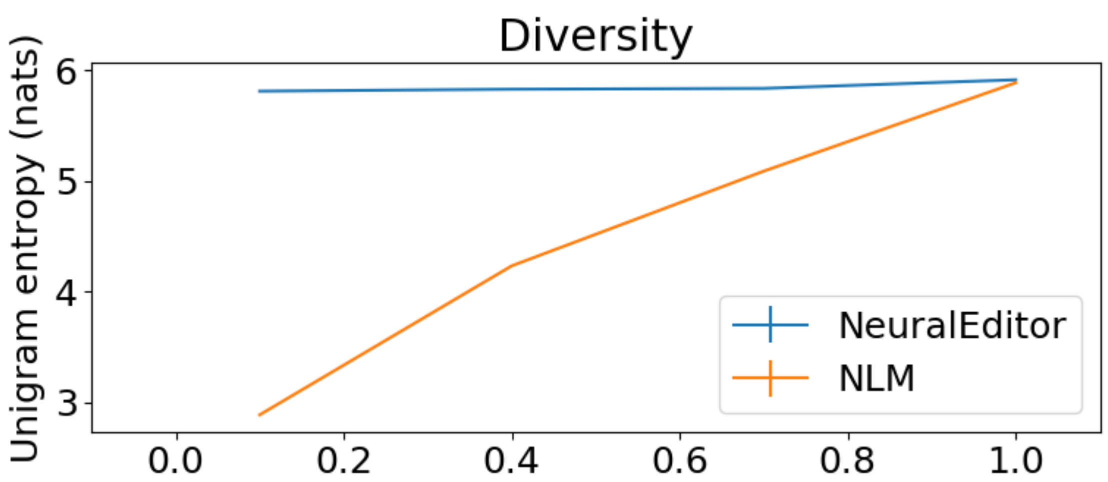


still grammatical

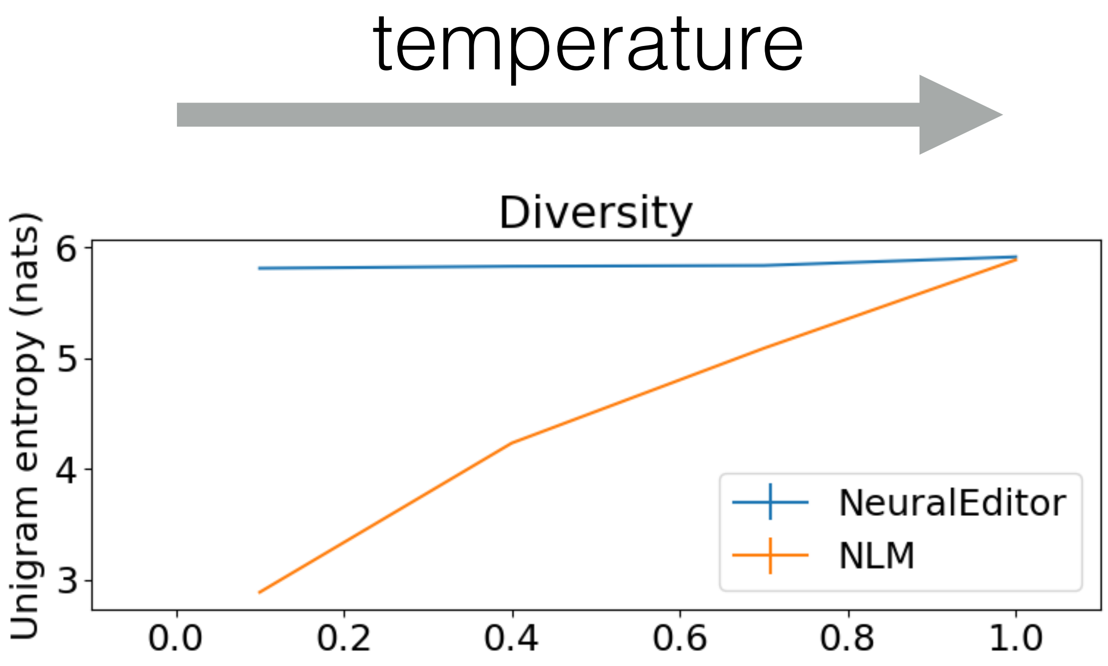
editor is only modeling minor variation
distribution is much easier to represent

Diversity: NLM vs NeuralEditor

Diversity: NLM vs NeuralEditor



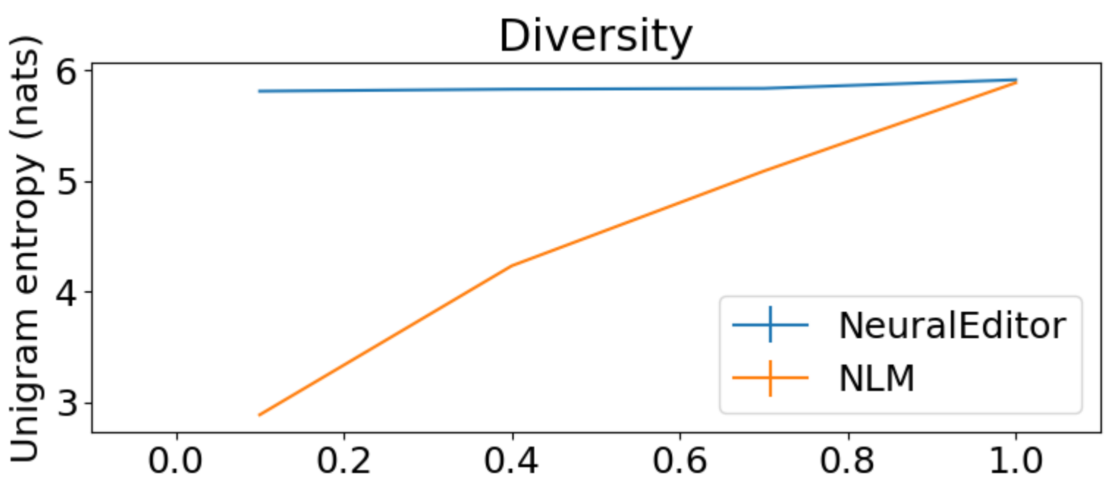
Diversity: NLM vs NeuralEditor



Diversity: NLM vs NeuralEditor

temperature

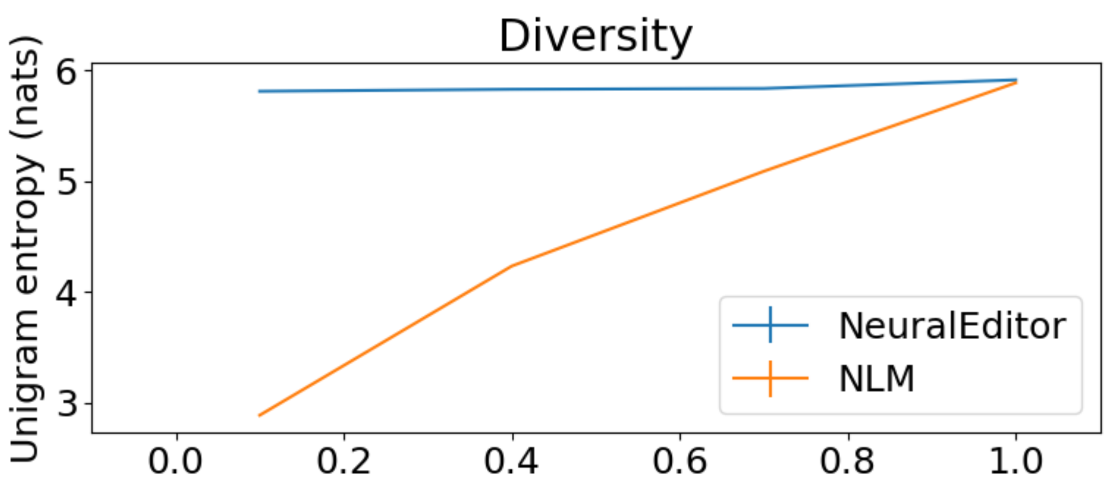
blue = NeuralEditor **orange** = NLM



Diversity: NLM vs NeuralEditor

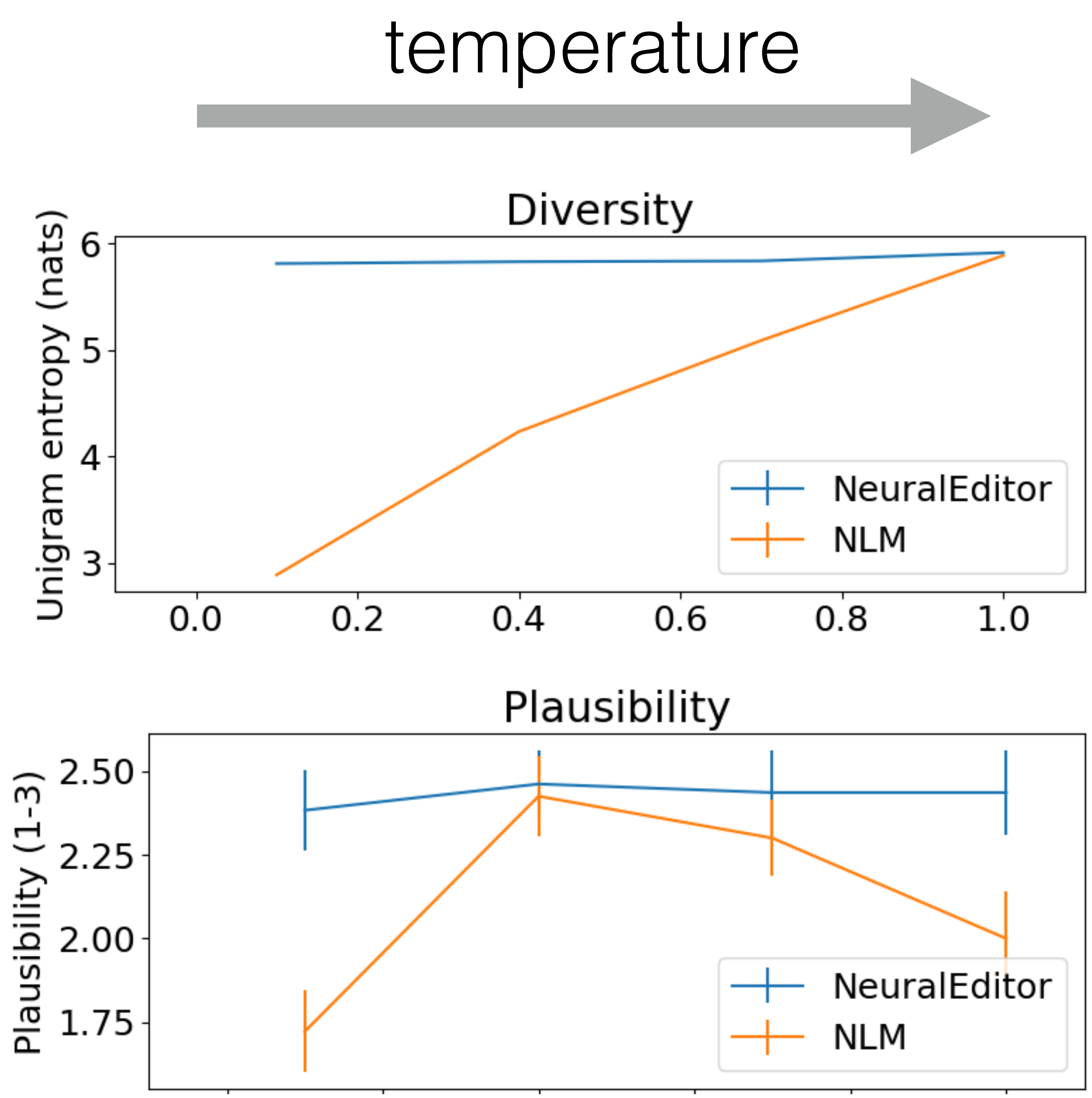
temperature

blue = NeuralEditor **orange** = NLM



NeuralEditor is always diverse even at temperature = 0

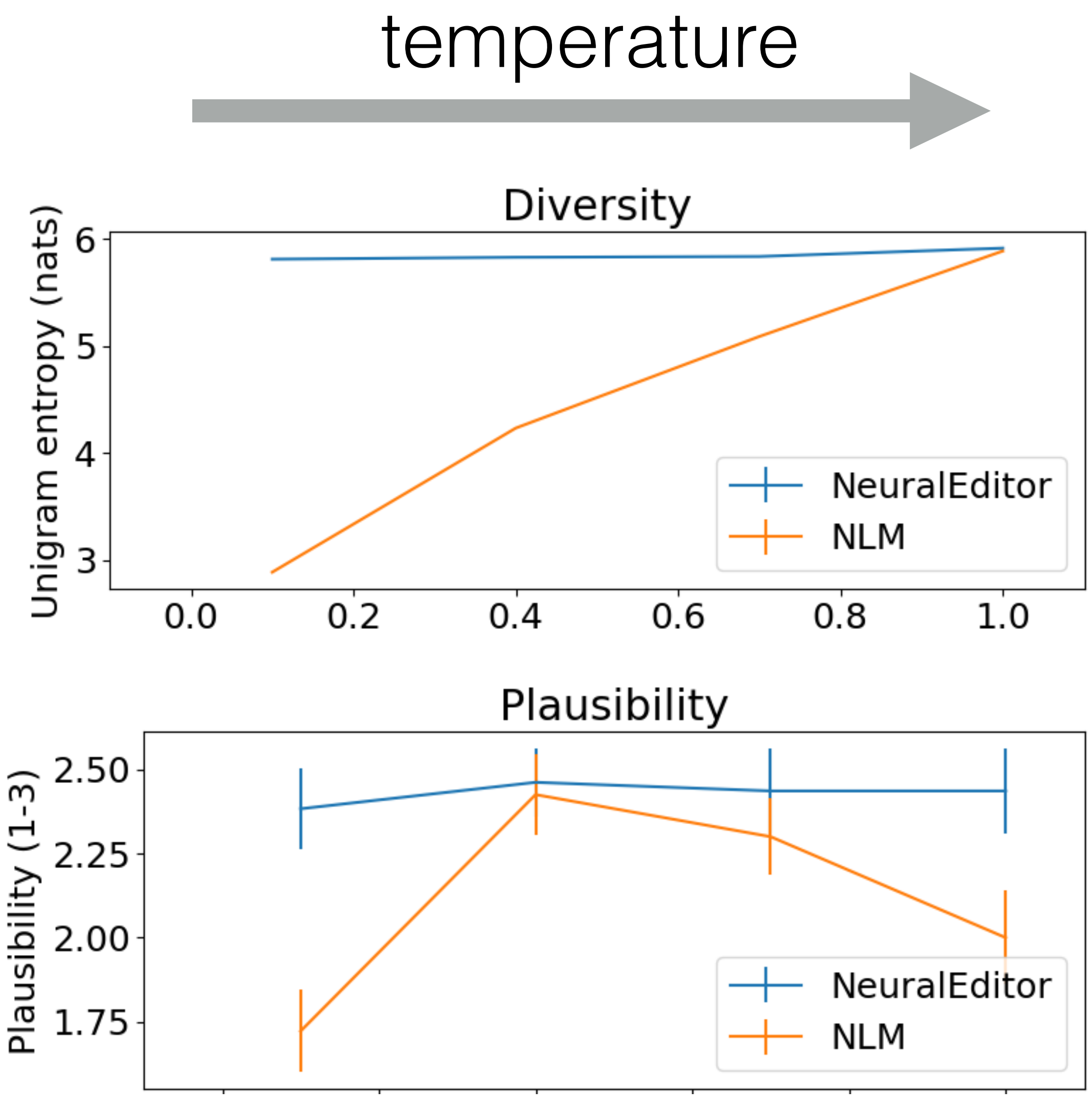
Diversity: NLM vs NeuralEditor



blue = NeuralEditor **orange** = NLM

NeuralEditor is always diverse even at temperature = 0

Diversity: NLM vs NeuralEditor

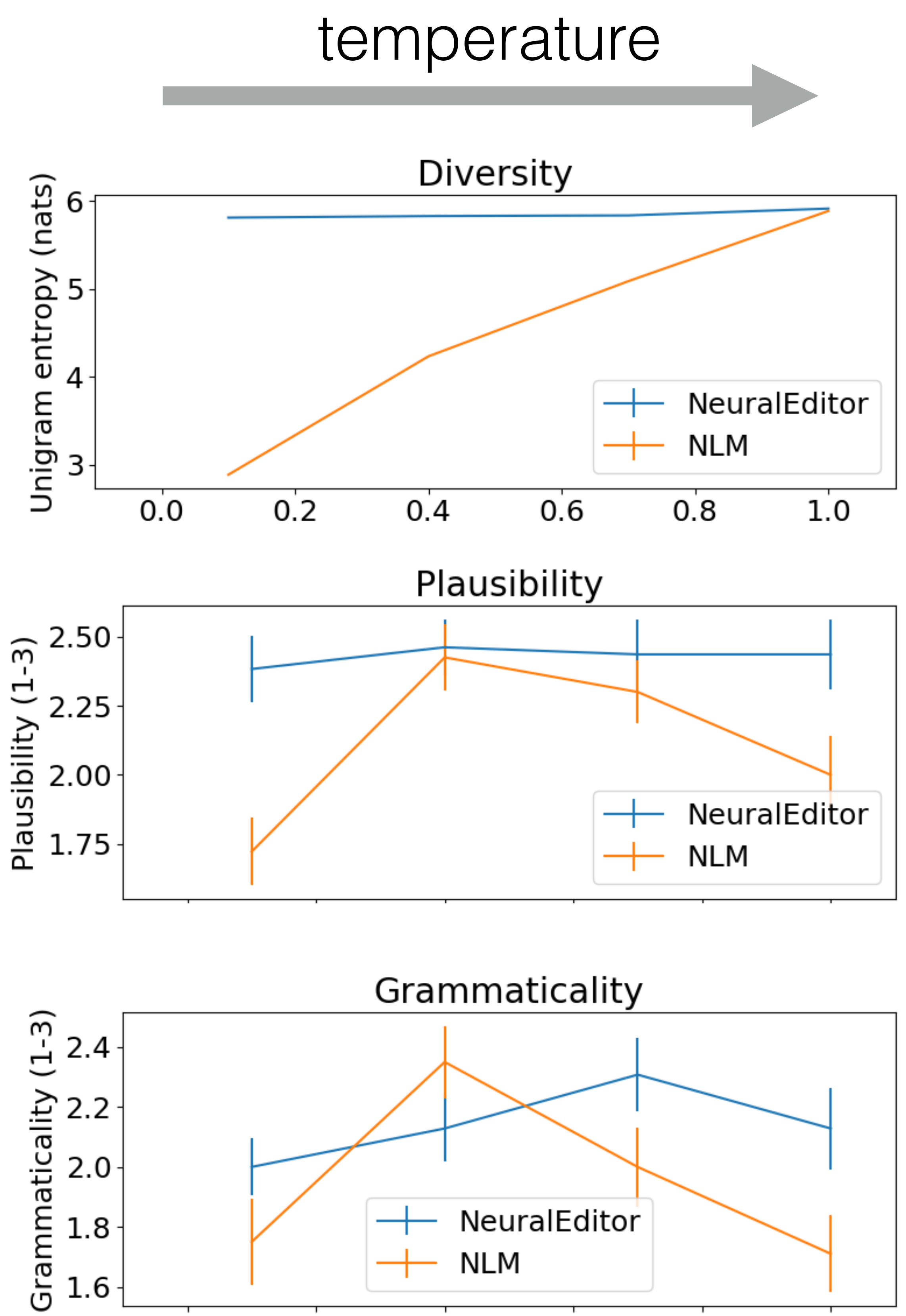


blue = NeuralEditor **orange** = NLM

NeuralEditor is always diverse even at temperature = 0

NeuralEditor generations more plausible at all temps

Diversity: NLM vs NeuralEditor

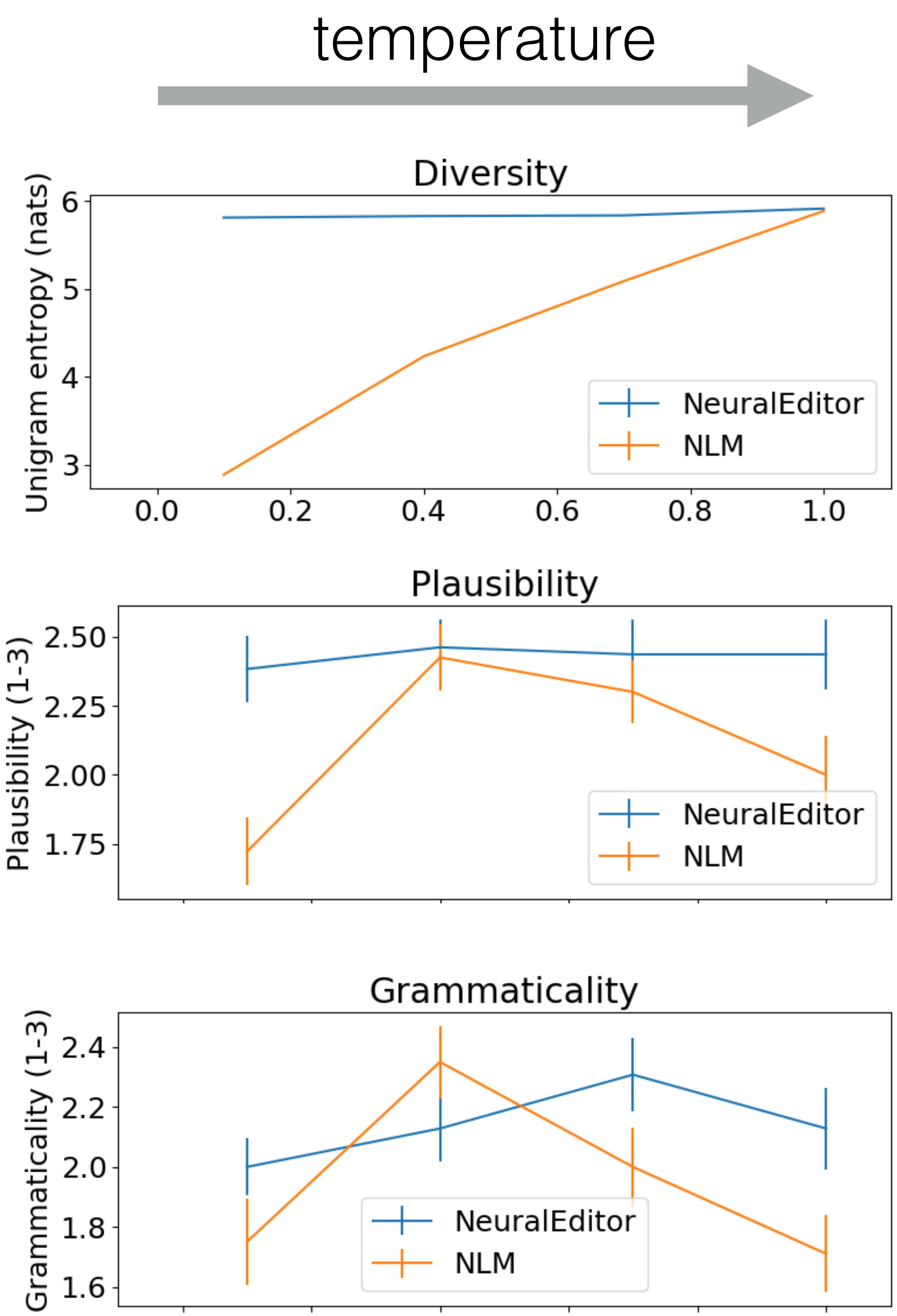


blue = NeuralEditor **orange** = NLM

NeuralEditor is always diverse even at temperature = 0

NeuralEditor generations more plausible at all temps

Diversity: NLM vs NeuralEditor



blue = NeuralEditor **orange** = NLM

NeuralEditor is always diverse even at temperature = 0

NeuralEditor generations more plausible at all temps

NLM grammaticality suffers for higher temperatures

Results

✓ **More diverse generations**

✓ **Higher quality generations**

✓ **Better perplexity** (BillionWord, Yelp reviews)

✓ **Edits are semantically meaningful**

- preserve semantic similarity
- can be used to perform sentence-level analogies

\end{**Results**}