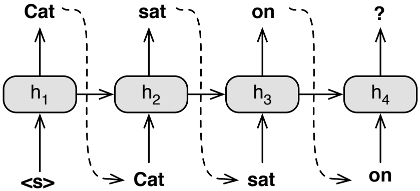
Deep Residual Output Layers for Neural Language Generation

Nikolaos Pappas, James Henderson

June 13, 2019

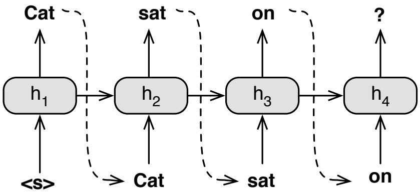
Neural language generation



Probability distribution at time t given context vector $h_t \in \mathbb{R}^d$, weights $W \in \mathbb{R}^{d \times |\mathcal{V}|}$ and bias $b \in \mathbb{R}^{|\mathcal{V}|}$:

$$p(y_t | y_1^{t-1}) \propto \exp(W^T h_t + b)$$

Neural language generation



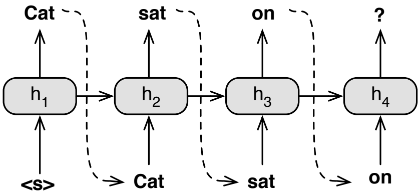
Probability distribution at time t given context vector $h_t \in \mathbb{R}^d$, weights $W \in \mathbb{R}^{d \times |\mathcal{V}|}$ and bias $b \in \mathbb{R}^{|\mathcal{V}|}$:

$$p(y_t | y_1^{t-1}) \propto \exp(W^T h_t + b)$$

- Output layer parameterisation depends on the vocabulary size $|\mathcal{V}|$

  $\rightarrow$ **Sample inefficient**

Neural language generation

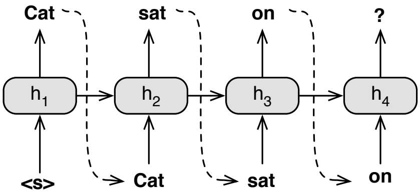


Probability distribution at time t given context vector $h_t \in \mathbb{R}^d$, weights $W \in \mathbb{R}^{d\times|\mathcal{V}|}$ and bias $b \in \mathbb{R}^{|\mathcal{V}|}$:

$$p(y_t|y_1^{t-1}) \propto \exp(W^T h_t + b)$$

- Output layer parameterisation depends on the vocabulary size $|\mathcal{V}|$
  $\rightarrow$ **Sample inefficient**
- Output layer power depends on hidden dim or rank d: "softmax bottleneck"
  $\rightarrow$ **High overhead and prone to overfitting**

Previous work



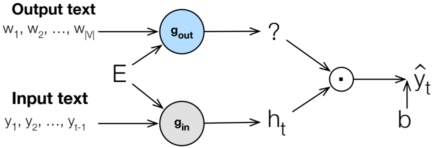
Probability distribution at time t given context vector $h_t \in \mathbb{R}^d$, weights $W \in \mathbb{R}^{d \times |\mathcal{V}|}$ and bias $b \in \mathbb{R}^{|\mathcal{V}|}$:

$$p(y_t|y_1^{t-1}) \propto \exp(W^T h_t + b)$$

- Output layer parameterisation no longer depends on the vocabulary size $|\mathcal{V}|$ (1)
  $\rightarrow$ **More sample efficient**
- Output layer power still depends on hidden dim or rank d: "softmax bottleneck" (2)
  $\rightarrow$ **High overhead and prone to overfitting**

Output similarity structure learning methods help with (1) but not yet with (2).

Previous work

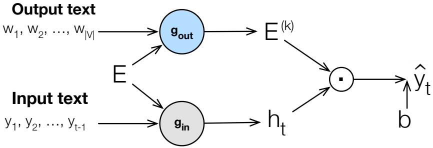


Output structure learning factorization of probability distribution given word embedding $E \in \mathbb{R}^{|\mathcal{V}| \times d}$:

$$p(y_t | y_1^{t-1}) \propto g_{out}(E, \mathcal{V}) g_{in}(E, y_1^{t-1}) + b$$

- Shallow label encoder networks such as weight tying [PW17], bilinear mapping [G18], and dual nonlinear mapping [P18]

Our contributions

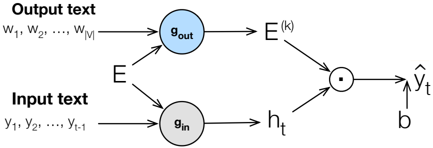


Output structure learning factorization of probability distribution given word embedding $E \in \mathbb{R}^{|\mathcal{V}| \times d}$:

$$p(y_t | y_1^{t-1}) \propto g_{out}(E, \mathcal{V}) g_{in}(E, y_1^{t-1}) + b$$

- Generalize previous output similarity structure learning methods
  $\rightarrow$ **More sample efficient**

Our contributions

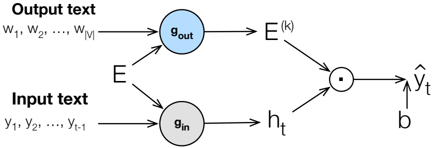


Output structure learning factorization of probability distribution given word embedding $E \in \mathbb{R}^{|\mathcal{V}| \times d}$:

$$p(y_t | y_1^{t-1}) \propto g_{out}(E, \mathcal{V}) g_{in}(E, y_1^{t-1}) + b$$

- Generalize previous output similarity structure learning methods
  $\rightarrow$ **More sample efficient**
- Propose a deep output label encoder network with dropout between layers
  $\rightarrow$ **Avoids overfitting**

Our contributions

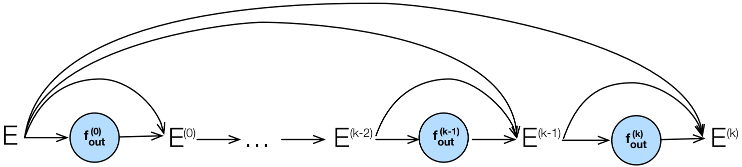


Output structure learning factorization of probability distribution given word embedding $E \in \mathbb{R}^{|\mathcal{V}| \times d}$:

$$p(y_t | y_1^{t-1}) \propto g_{out}(E, \mathcal{V}) g_{in}(E, y_1^{t-1}) + b$$

- Generalize previous output similarity structure learning methods
  - $\rightarrow$ **More sample efficient**
- Propose a deep output label encoder network with dropout between layers
  - $\rightarrow$ **Avoids overfitting**
- Increase output layer power with representation depth instead of rank d
  - $\rightarrow$ **Low overhead**

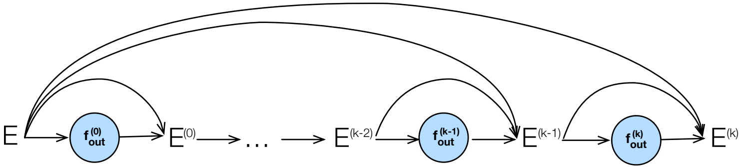
Label Encoder Network



- Shares parameters across output labels with k nonlinear projections

$$E^{(k)} = f_{out}^{(k)}(E^{(k-1)})$$

Label Encoder Network



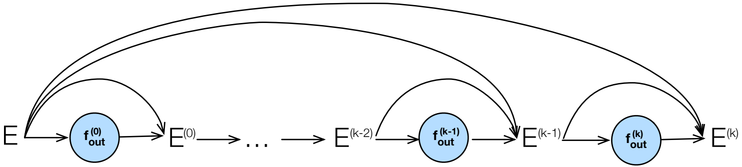
- Shares parameters across output labels with k nonlinear projections

$$E^{(k)} = f_{out}^{(k)}(E^{(k-1)})$$

- Preserves information across layers with residual connections

$$E^{(k)} = f_{out}^{(k)}(E^{(k-1)}) + E^{(k-1)} + E$$

Label Encoder Network



- Shares parameters across output labels with k nonlinear projections

$$E^{(k)} = f_{out}^{(k)}(E^{(k-1)})$$

- Preserves information across layers with residual connections

$$E^{(k)} = f_{out}^{(k)}(E^{(k-1)}) + E^{(k-1)} + E$$

- Avoids overfitting with standard or variational dropout for each layer $i = 1, \ldots, k$

$$f'^{(i)}_{out}(E^{(i-1)}) = \delta\left(f_{out}^{(i)}(E^{(i-1)})\right) \odot f_{out}^{(i)}(E^{(i-1)})$$

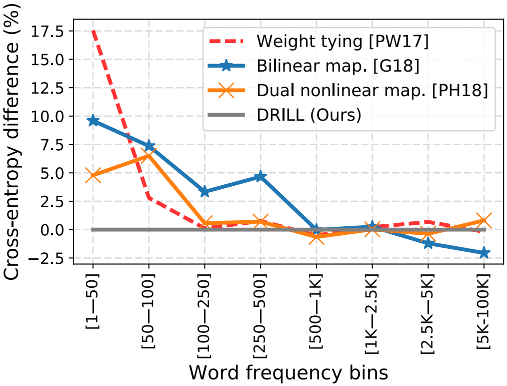
Results

- Improve competitive architectures without increasing their dim or rank

Language modeling	ppl	sec/ep
AWD-LSTM [M18]	65.8	89 (1.0×)
AWD-LSTM-DRILL	61.9	106 (1.2×)
AWD-LSTM-MoS [Y18]	61.4	862 (9.7×)

Machine translation	bleu	min/ep
Transformer [V17]	27.3	111 (1.0×)
Transformer-DRILL	28.1	189 (1.7×)
Transformer (big) [V17]	28.4	779 (7.0×)

- Better transfer across low-resource output labels



Talk to us at **Poster #104 in Pacific Ballroom**.

Thank you!

`http://github.com/idiap/drill`