Data Selection

付振新

2019.9.16

Survey of Data-Selection Methods in Statistical Machine Translation

Sauleh Eetemadi · William Lewis

Kristina Toutanova · Hayder Radha

Main Characteristics

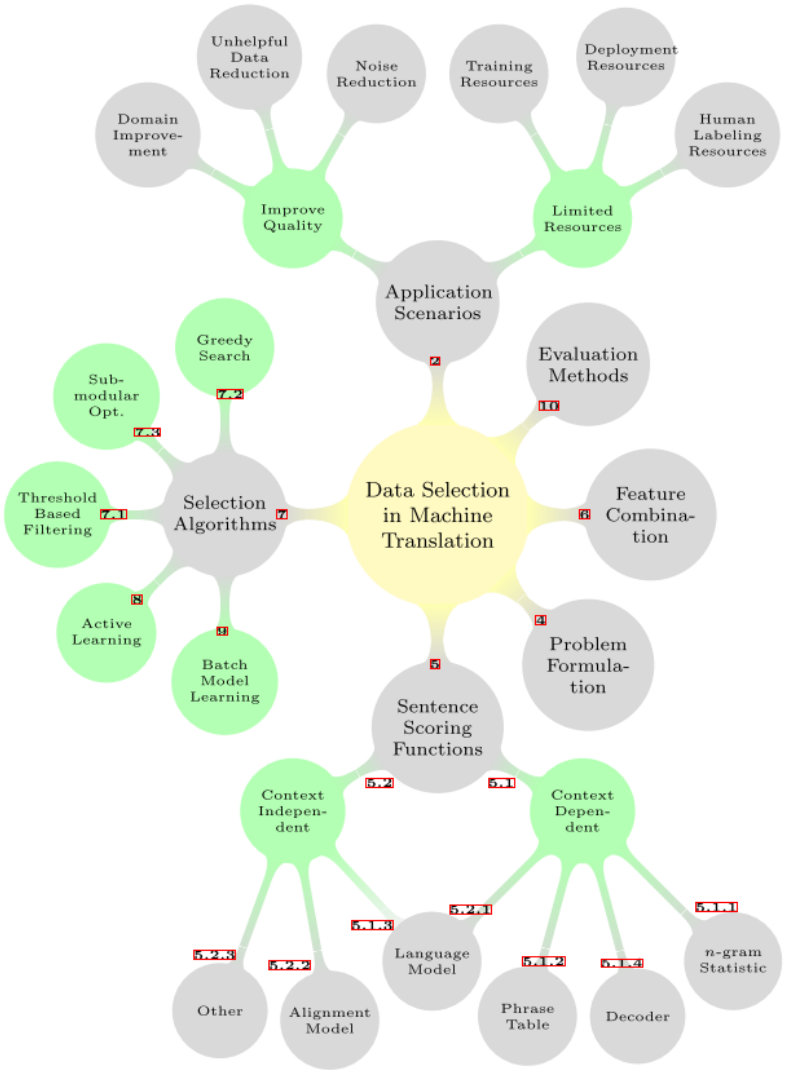
- **Application Scenario**: The application scenario for each method defines the type of data that is being selected (parallel or monolingual), and the problem that the method aims to solve.
- **Scoring Functions**: Scoring functions (features) are expected to correlate with the usefulness of the data subset for training a high-quality MT system.
- **Selection Algorithms**: Finding the subset that maximizes this scoring function subject to some constraints.

Main Characteristics

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \sum_{i=1}^{\substack{Feature \\ Count}} \lambda_i \ \mathcal{F}_i(S)$$

$\mathcal{F}_i(S)$: corpus-level scoring function

λ_i : feature weight



Application Scenarios

- Satisfying Resource Constraints
  - Training Resources
  - Deployment Size
  - Manual Translation Cost
- Quality Improvement
  - Noise reduction
  - Reduction of Unhelpful Data
  - Domain Improvement

Sentence-Level Scoring Functions

- Context-Independent
  - which depend on nothing but the candidate sentences.
- Context-dependent
  - which depend on the selected pool

Context-dependent Functions

N-Gram Coverage Functions

$$f_1(s_j|S_1^{j-1}) = \frac{1}{norm(s_j)} \sum_{n=1}^{N} \sum_{ng \in \mathrm{NG}(s_j,n)} weight(ng, j-1)$$

$\mathrm{NG}(s,n)$: All n-grams of size n in sentence s

$weight(ng,m)$: Weight of n-gram ng given selected pool S_1^m are selected

$norm(s)$: Normalization factor for sentence s

$$weight_1(ng,m) = \begin{cases} 0 & ng \in \bigcup_{i=1}^{m} \mathrm{NG}(s_m,n) \\ 1 & \text{otherwise} \end{cases}$$

$$norm_1(s) = |\,\mathrm{NG}(s,n)|$$

Phrase-Table-Based Functions

- Phrase pairs that occur frequently in the selection pool but do not occur (or occur rarely) in the seed corpus are of most value.

Decoder-Based Functions

- Based on the error or the distance between the produced translation and the target side of the sentence pair.
- The idea is that if the SMT system makes an error on a sentence, the sentence is likely to contain useful information.
- **Round Trip Translation Accuracy**
  - 先正向翻译，再反向翻译，看两次翻译后和原句的差异最大的样本加入训练数据
- **Inter-System Disagreement**
  - 训练多个模型，模型分歧最大的样本加入训练数据

One More

- Estimate the contribution of each sentence pair in the selection pool by sampling a large number of random subsets of the selection pool with replacement and training an SMT system over each subset of the training data.（采样很多个subset，之后对每个subset训练SMT模型，之后评估模型效果。）
- After calculating the performance of each system (using BLEU score, say), a contribution score is estimated for each sentence pair based on its membership in different subsets and the performance of those subsets.（每个pair的得分是它所属的subset的得分的整合）

Context-Independent Functions

Language Model-Based Functions

- Denkowski et al (2012) use an external source and target language model trained on clean data to filter out ungrammatical sentences.
- Yasuda et al (2008) use a language model trained on an in-domain development set to score sentences in the selection pool
- Moore and Lewis address this issue by introducing a second language model trained on a general-domain development set. In their approach, they use the cross-entropy diff
$$f_7(s_i|S_{\text{in}}, S_{\text{out}}) = H^{ind}_{\mathcal{LM}(S_{\text{in}})}(s_i) - H^{ind}_{\mathcal{LM}(S_{\text{out}})}(s_i)$$
language models to score sentences

Dynamically Composing Domain-Data Selection with Clean-Data Selection by "Co-Curricular Learning" for Neural Machine Translation

2.1 Measuring Domain and Noise in Data

Data selection for MT usually uses a scoring function to rank sentence pairs. Cross entropy difference (Moore and Lewis, 2010) between two language models is usually used for selecting domain sentences, e.g., (van der Wees et al., 2017; Axelrod et al., 2011). For a source sentence x of length $|x|$, with a general-domain language model (LM), parameterized as $\widetilde{\vartheta}$, and an in-domain LM, $\widehat{\vartheta}$, the domain-relevance of x is calculated as:[1]

$$\varphi\left(x;\widetilde{\vartheta},\widehat{\vartheta}\right) = \frac{\log P\left(x;\widehat{\vartheta}\right) - \log P\left(x;\widetilde{\vartheta}\right)}{|x|} \quad (1)$$

Data selection has also been used for data denoising (Junczys-Dowmunt, 2018; Wang et al., 2018b), by using NMT models and trusted data to measure the noise level in a sentence pair. One such a scoring function uses a baseline NMT, $\widetilde{\theta}$, trained on noisy data and a cleaner NMT, $\widehat{\theta}$, obtained by fine-tuning $\widetilde{\theta}$ on a small trusted parallel dataset, and measures quality in a sentence pair (x, y):

$$\phi\left(x,y;\widetilde{\theta},\widehat{\theta}\right) = \frac{\log P\left(y|x;\widehat{\theta}\right) - \log P\left(y|x;\widetilde{\theta}\right)}{|y|} \quad (2)$$

Others

- Alignment Model-Based Functions
  - These scoring functions attempt to quantify the translation quality and accuracy of the sentence pair. Any word alignment model can be used to compute this scoring function by aligning each word in a sentence to corresponding word(s) in its pair.
- Length-based functions
  - Source sentence to target sentence length ratio and vice
  - Difference of source and target sentence lengths
  - Sentence length
  - Length of longest token in the sentence

Feature combination and parameter tuning

- A common requirement for parameter tuning and feature combination is the availability of a development set.
- In a supervised learning scenario, human annotators are used to create the development set.
- Since this is an expensive and time-consuming task, alternative techniques for development data creation in semi-supervised learning methods have been developed.

Selection-Based Filtering

- Threshold-Based Filtering
- Greedy Search
- Submodular Optimization

Submodular Optimization (次模优化)

就是对于一个集合函数 $f: 2^V \rightarrow R$

，若 $S \subseteq V$，那么在S中增加一个元素所增加的收益要小于等于在S的子集中增加一个元素所增加的收益。

形式化表述就是：对于函数f而言，若 $A \subseteq B \subseteq V$ 且 $e \in V - B$，则
$f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$

通俗的说就是你把所有商品看成一个集合，随着你所拥有的物品数量的增加，那么你获得剩下物品所得到的满足程度越来越小。

在计算机领域，Submodular函数有个性质，即下面这个问题是NP-hard.

对于一个sugmodular函数f，如果给定一个限制条件C，找出一个满足条件C的集合S，使得f(S)值最大。

进一步而言，对于上述NP-hard问题，最基本的贪心算法是每次迭代地在解中加入增益最大且满足条件C的元素，即第i次迭代时解 $S_i = S_{i-1} \cup \{argmax_e \Delta(e|S_{i-1})\}$，其中
$\Delta(e|S_{i-1}) = f(S_{i-1} \cup \{e\}) - f(S_{i-1})$

对于如果一个submodular function f是单调且非负的，即对于任意 $e \notin S$，
$f(S \cup \{e\}) \geq f(S)$ 且对于任意 $S \subseteq V$，$f(S) \geq 0$，那么上面的贪心算法找出的近似解 S_{app} 相对于最优解 S_{opt} 满足 $f(S_{app}) \geq (1 - \frac{1}{e}) f(S_{opt})$

Submodular Optimization

1. U: This is the set of features present in all sentences in the selection pool. N-grams are a natural choice for this.
2. $m_u(x)$: This function calculates the relevance of each feature u to sentence x. Kirchhoff and Bilmes use TF-IDF for this function. The main constraint with this function is that it has to be modular.
3. $w(u)$: Each feature can have a weight function to present its importance. In a domain-adaptation task, the frequency of the n-gram in an in-domain development set can be used for this weight function.
4. $\phi_u(a)$: This decay function determines the rate of diminishing returns for redundant instances of a feature u. As this concave function becomes flat, the feature loses its ability to provide additional improvement to the value of a candidate subset. Kirchhoff and Bilmes experiment with square root and logarithmic functions.

Active Learning

- Active learning is a technique used in machine learning where the algorithm can choose the new data from which it learns
  - Pool-based: The learning algorithm selects data points from a pool of unlabeled data.
  - Stream-based: All available unlabeled data is streamed through the learning algorithm and the learning algorithm can choose to query for the data point or discard.
  - Query Synthesis: The learning algorithm generates new data points to be labeled by the oracle.
- Active learning selects new data points in an iterative process. In each step, it selects new data point(s), queries the oracle for the label and learns from the results.

Batch-Mode Learning

- Batch-mode learning is applicable to both active learning as well as data selection for passive learning.
- The idea is to select new data points in batches rather than one at a time.

Summary

- In practice, performing a data-selection task for SMT requires addressing two technical questions:
  - Scoring Function: How is the value of a sentence or sentence pair objectively evaluated?
  - Selection Algorithm: Given a scoring function and a constraint, how is the optimum data subset selected?

More……

- Instance Weighting
  - Learning to Converse with Noisy Data: Generation with Calibration
  - Not all dialogues are created equal: Instance weighting for neural conversational models
  - Avoiding Your Teacher’s Mistakes: Training Neural Networks with Controlled Weak Supervision

Improving Neural Conversational Models with Entropy-Based Data Filtering

Method

IDENTITY

The probabilities are based on the observed relative frequency of utterance pairs in the data.

$$H_{\text{tgt}}(s, D) = -\sum_{(s,t_i)\in D} p(t_i|s)\log_2 p(t_i|s)$$

$$H_{\text{src}}(t, D) = -\sum_{(s_i,t)\in D} p(s_i|t)\log_2 p(s_i|t)$$

AVG-EMBEDDING
SENT2VEC

To this end we performed the filtering based not only on the set of all utterances, as in the case of IDENTITY, but also on clusters of utterances established by clustering their vector representations using the Mean Shift algorithm.

$$H_{\text{tgt}}(c_s, C) = -\sum_{c_i\in C} p(c_i|c_s)\log_2 p(c_i|c_s)$$

$$H_{src}\ (C_t, C) = -\sum_{c_i\in C} p(c_i|c_t)\log_2 p(c_i|c_t)$$

Analysis

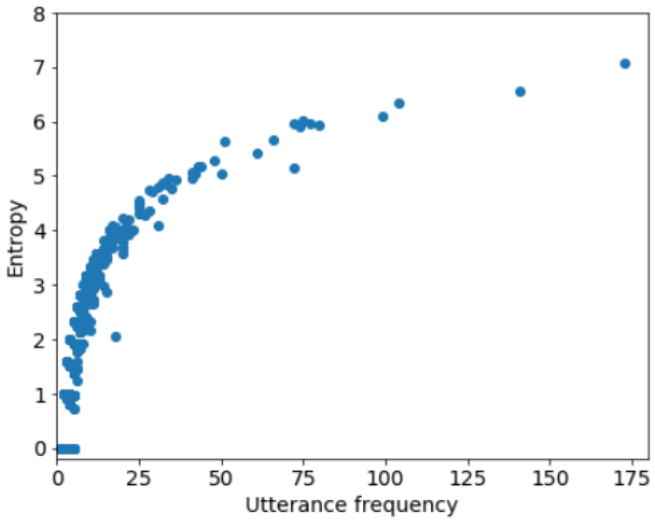


Figure 1: Entropy of source utterances (computed with IDENTITY) with respect to utterance frequency.

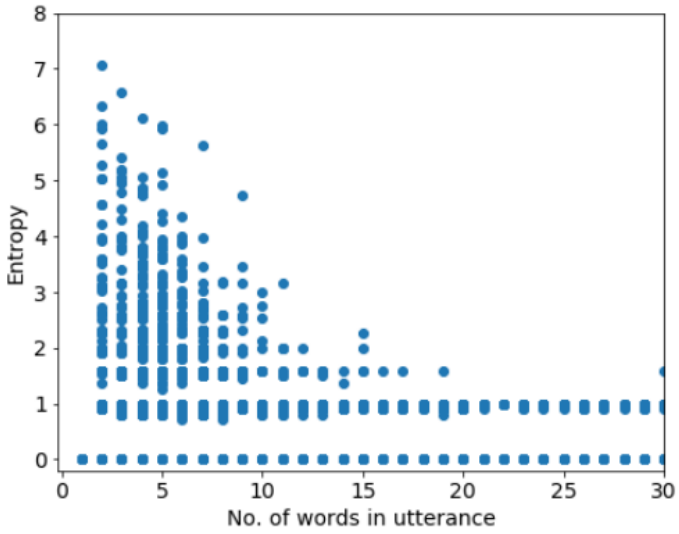


Figure 2: Entropy of source utterances (computed with IDENTITY) with respect to utterance length.

Cluster

```
hi  an  .  how are you  ?
hi  craig  !  how are you  ?
hi  how are you  .  is alice there  ?
hi  !  how are you doing  ?
hi  francis  morning  !  how are you doing today  ?
hi  peter  !  how are you  ?
hi  randy  .  what are you doing right now  ?
hi  jane  .  how are you doing this morning  ?
hi  nancy  .  how are you doing  ?
hi  how are you doing  ?
hi  nancy . how are you doing  ?
hi steve  .  this is mike  .  what are you doing  ?
hi  how are you  ?
hi b  .  how are you  ?
hi  alex  .  how are you doing  ?
hi  !  how are you going  ?
hi mike  how are you doing  ?
hi  .  how can i help you  ?
hi  jack  !  how are you doing  ?
hi  carlos  .  what are you doing this afternoon  ?
hi  victor  .  how are you  ?
oh  yes  .  hi  how are you  ?
hi  tom  .  how have you been  ?
hi  bob  !  how are you doing  ?
hi  alice . how are you  ?
hi  brad  .  how are you today  ?
```

Figure 3: A cluster produced by SENT2VEC.

Experiments

Dataset	Type	Th.	SOURCE	TARGET	BOTH
	ID	1	5.64%	6.98%	12.2%
DailyDialog	AE	3.5	5.39%	7.06%	12.0%
	SC	3.5	6.53%	8.45%	14.3%
Cornell	ID	4	-	7.39%	14.1%
Twitter	ID	0.5	-	1.82%	9.96%

Table 1: Entropy threshold (Th.) and amount of data filtered for all datasets in the 3 filtering scenarios. ID stands for IDENTITY, AE stands for AVG-EMBEDDING, and SC for SENT2VEC.

Results

		$\|U\|$	H_w^u	H_w^b	H_u^u	H_u^b	D_{kl}^u	D_{kl}^b	AVG	EXT	GRE	COH	d1	d2	b1	b2	b3	b4
TRF		8.6	7.30	12.2	63.6	93	.330	.85	.540	.497	.552	.538	**.0290**	.149	.142	.135	.130	.119
ID	B	9.8	7.44	12.3	71.9	105	.315	.77	.559	**.506**	*.555*	.572	.0247	.138	.157	.151	.147	.136
	T	*10.9*	**7.67**	**12.7**	**83.2**	**121**	**.286**	**.72**	**.570**	**.507**	.554	**.584**	.0266	**.150**	**.161**	**.159**	**.156**	**.146**
	S	9.4	7.19	11.9	66.4	98	.462	1.08	.540	.495	.553	.538	.0262	.130	.139	.133	.128	.117
AE	B	7.9	7.25	12.0	57.7	83	.447	1.05	.524	.486	.548	.524	.0283	.132	.128	.121	.115	.105
	T	8.6	7.26	12.1	61.4	90	.425	1.12	.526	.492	.548	.529	.0236	.115	.133	.127	.121	.111
	S	*9.0*	7.21	11.9	*65.1*	*95*	.496	1.16	.536	.490	.548	.538	.0232	.109	.134	.130	.126	.116
SC	B	10.0	7.40	12.3	72.6	108	.383	.97	.544	.497	.549	.550	.0257	.131	.145	.142	.138	.128
	T	**11.2**	*7.49*	*12.4*	**82.2**	**122**	.391	.97	*.565*	*.500*	.552	*.572*	.0250	.132	*.153*	*.153*	*.152*	*.142*
	S	**11.1**	7.15	11.9	74.4	114	.534	1.27	.546	*.501*	**.560**	.544	.0213	.102	.144	.139	.135	.125

Table 2: Metrics computed at the minimum of the validation loss on the unfiltered test set (DailyDialog). TRF refers to `transformer`, **ID** to IDENTITY, **AE** to AVG-EMBEDDING, and **SC** to SENT2VEC. SOURCE-side, TARGET-side, and filtering BOTH sides are denoted by initials. Best results are highlighted with bold and best results separately for each entropy computing method are in italic (and those within a 95% confidence interval).

A small finding

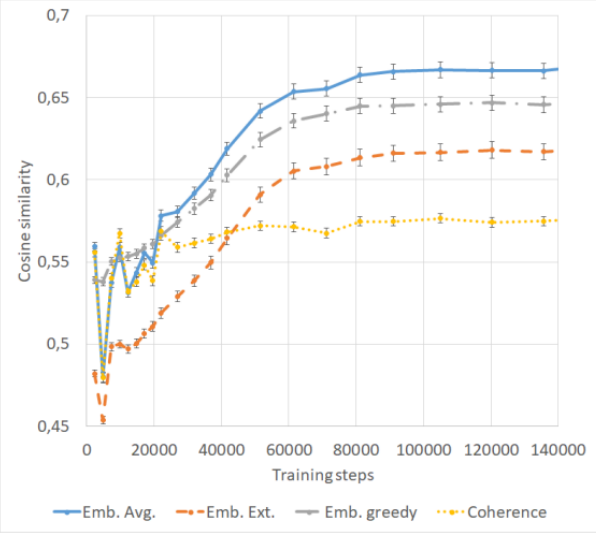


Figure 4: Embedding metrics and coherence (on validation data) as a function of the training evolution of `transformer` on unfiltered data (DailyDialog).

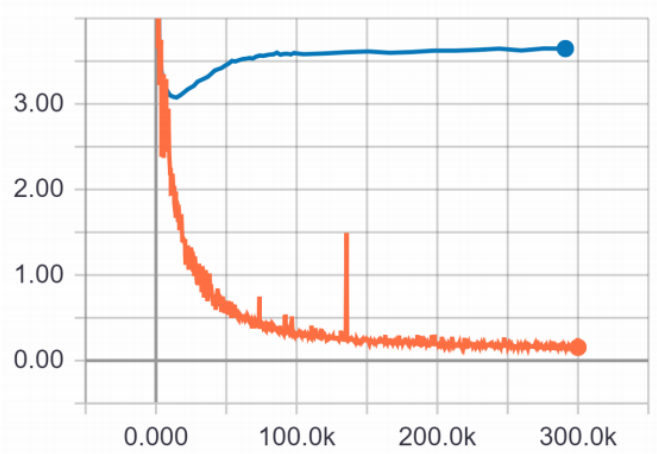


Figure 5: Training (bottom) and validation (top) loss with respect to training steps of `transformer` trained on unfiltered data (DailyDialog).

One more……

Do Neural Dialog Systems Use the Conversation History Effectively? An Empirical Study

Chinnadhurai Sankar[1,2,4*] **Sandeep Subramanian**[1,2,5]

Christopher Pal[1,3,5] **Sarath Chandar**[1,2,4] **Yoshua Bengio** [1,2]

[1]Mila [2]Université de Montréal [3]École Polytechnique de Montréal
[4]Google Research, Brain Team [5]Element AI, Montréal

ACL 2019 Short

Perplexity Change

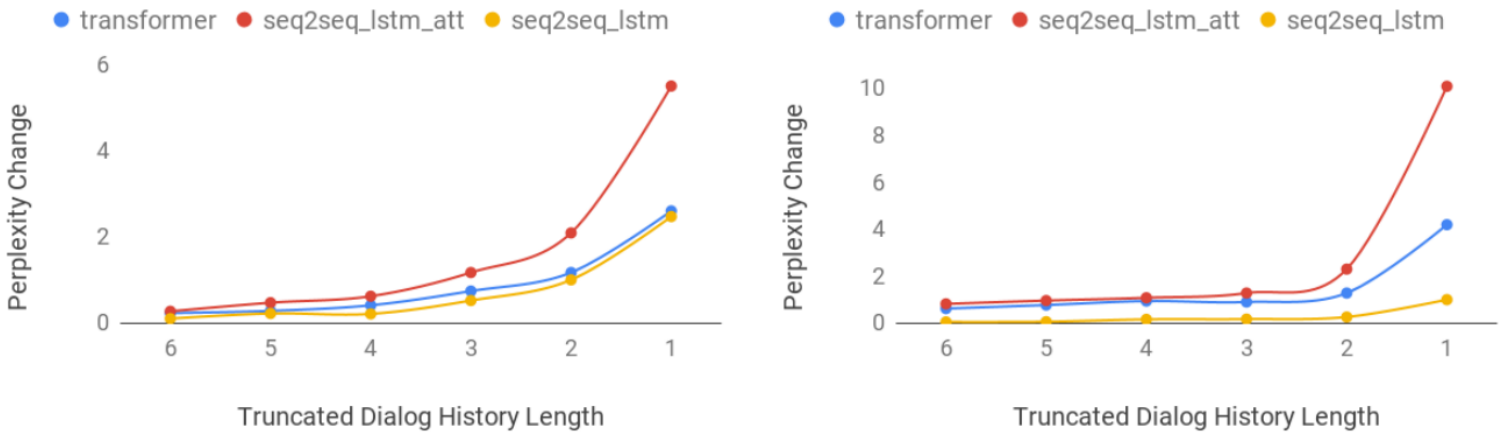


Figure 1: The increase in perplexity for different models when only presented with the k most recent utterances from the dialog history for Dailydialog (left) and bAbI dialog (right) datasets. Recurrent models with attention fare better than transformers, since they use more of the conversation history.

Results

Models	Test PPL	Only Last	Shuf	Rev	Drop First	Drop Last	Word Drop	Verb Drop	Noun Drop	Word Shuf	Word Rev
		Utterance level perturbations ($\Delta PPL_{[\sigma]}$)					Word level perturbations ($\Delta PPL_{[\sigma]}$)				
DailyDialog											
seq2seq_lstm	$32.90_{[1.40]}$	$1.70_{[0.41]}$	$\mathbf{3.35}_{[0.38]}$	$\mathbf{4.04}_{[0.28]}$	$0.13_{[0.04]}$	$\mathbf{5.08}_{[0.79]}$	$1.58_{[0.15]}$	$0.87_{[0.08]}$	$1.06_{[0.28]}$	$\mathbf{3.37}_{[0.33]}$	$3.10_{[0.45]}$
seq2seq_lstm_att	$29.65_{[1.10]}$	$\mathbf{4.76}_{[0.39]}$	$2.54_{[0.24]}$	$3.31_{[0.49]}$	$\mathbf{0.32}_{[0.03]}$	$4.84_{[0.42]}$	$\mathbf{2.03}_{[0.25]}$	$\mathbf{1.37}_{[0.29]}$	$2.22_{[0.22]}$	$2.82_{[0.31]}$	$\mathbf{3.29}_{[0.25]}$
transformer	$\mathbf{28.73}_{[1.30]}$	$3.28_{[1.37]}$	$0.82_{[0.40]}$	$1.25_{[0.62]}$	$0.27_{[0.19]}$	$2.43_{[0.83]}$	$1.20_{[0.69]}$	$0.63_{[0.17]}$	$\mathbf{2.60}_{[0.98]}$	$0.15_{[0.08]}$	$0.26_{[0.18]}$
Persona Chat											
seq2seq_lstm	$43.24_{[0.99]}$	$3.27_{[0.13]}$	$6.29_{[0.48]}$	$\mathbf{13.11}_{[1.22]}$	$0.47_{[0.21]}$	$\mathbf{6.10}_{[0.46]}$	$1.81_{[0.25]}$	$0.68_{[0.19]}$	$0.75_{[0.15]}$	$1.29_{[0.17]}$	$1.95_{[0.20]}$
seq2seq_lstm_att	$42.90_{[1.76]}$	$\mathbf{4.44}_{[0.81]}$	$\mathbf{6.70}_{[0.67]}$	$11.61_{[0.75]}$	$\mathbf{2.99}_{[2.24]}$	$5.58_{[0.45]}$	$\mathbf{2.47}_{[0.67]}$	$\mathbf{1.11}_{[0.27]}$	$\mathbf{1.20}_{[0.23]}$	$\mathbf{2.03}_{[0.46]}$	$\mathbf{2.39}_{[0.31]}$
transformer	$\mathbf{40.78}_{[0.31]}$	$1.90_{[0.08]}$	$1.22_{[0.22]}$	$1.41_{[0.54]}$	$-0.1_{[0.07]}$	$1.59_{[0.39]}$	$0.54_{[0.08]}$	$0.40_{[0.00]}$	$0.32_{[0.18]}$	$0.01_{[0.01]}$	$0.00_{[0.06]}$
MutualFriends											
seq2seq_lstm	$14.17_{[0.29]}$	$1.44_{[0.86]}$	$\mathbf{1.42}_{[0.25]}$	$1.24_{[0.34]}$	$0.00_{[0.00]}$	$0.76_{[0.10]}$	$0.28_{[0.11]}$	$0.00_{[0.03]}$	$0.61_{[0.39]}$	$0.31_{[0.25]}$	$0.56_{[0.39]}$
seq2seq_lstm_att	$\mathbf{10.60}_{[0.21]}$	$\mathbf{32.13}_{[4.08]}$	$1.24_{[0.19]}$	$1.06_{[0.24]}$	$0.08_{[0.03]}$	$\mathbf{1.35}_{[0.15]}$	$\mathbf{1.56}_{[0.20]}$	$0.15_{[0.07]}$	$\mathbf{3.28}_{[0.38]}$	$\mathbf{2.35}_{[0.22]}$	$\mathbf{4.59}_{[0.46]}$
transformer	$10.63_{[0.03]}$	$20.11_{[0.67]}$	$1.06_{[0.16]}$	$\mathbf{1.62}_{[0.44]}$	$\mathbf{0.12}_{[0.03]}$	$0.81_{[0.09]}$	$0.75_{[0.05]}$	$\mathbf{0.16}_{[0.02]}$	$1.50_{[0.12]}$	$0.07_{[0.01]}$	$0.13_{[0.04]}$
bAbi dailog: Task5											
seq2seq_lstm	$1.28_{[0.02]}$	$1.31_{[0.50]}$	$\mathbf{43.61}_{[15.9]}$	$\mathbf{40.99}_{[9.38]}$	$0.00_{[0.00]}$	$4.28_{[1.90]}$	$0.38_{[0.11]}$	$0.01_{[0.00]}$	$0.10_{[0.06]}$	$0.09_{[0.02]}$	$0.42_{[0.38]}$
seq2seq_lstm_att	$\mathbf{1.06}_{[0.02]}$	$\mathbf{9.14}_{[1.28]}$	$41.21_{[8.03]}$	$34.32_{[10.7]}$	$0.00_{[0.00]}$	$\mathbf{6.75}_{[1.86]}$	$\mathbf{0.64}_{[0.07]}$	$0.03_{[0.03]}$	$0.22_{[0.04]}$	$\mathbf{0.25}_{[0.01]}$	$\mathbf{1.10}_{[0.80]}$
transformer	$1.07_{[0.00]}$	$4.06_{[0.33]}$	$0.38_{[0.02]}$	$0.62_{[0.02]}$	$0.00_{[0.00]}$	$0.21_{[0.02]}$	$0.36_{[0.02]}$	$\mathbf{0.25}_{[0.06]}$	$\mathbf{0.37}_{[0.06]}$	$0.00_{[0.00]}$	$0.00_{[0.00]}$

Table 2: Model performance across multiple datasets and sensitivity to different perturbations. Columns 1 & 2 report the test set perplexity (without perturbations) of different models. Columns 3-12 report the **increase** in perplexity when models are subjected to different perturbations. The mean (μ) and standard deviation [σ] across 5 runs are reported. The *Only Last* column presents models with **only** the last utterance from the dialog history. The model that exhibits the highest sensitivity (higher the better) to a particular perturbation on a dataset is in bold. *seq2seq_lstm_att* are the most sensitive models **24/40** times, while transformers are the least with **6/40** times.

Thanks !