Learned in Translation: Contextualized Word Vectors

Bryan McCann, James Bradbury, Caiming Xiong, Richard Socher

NIPS 2017

Context Vectors (CoVe)

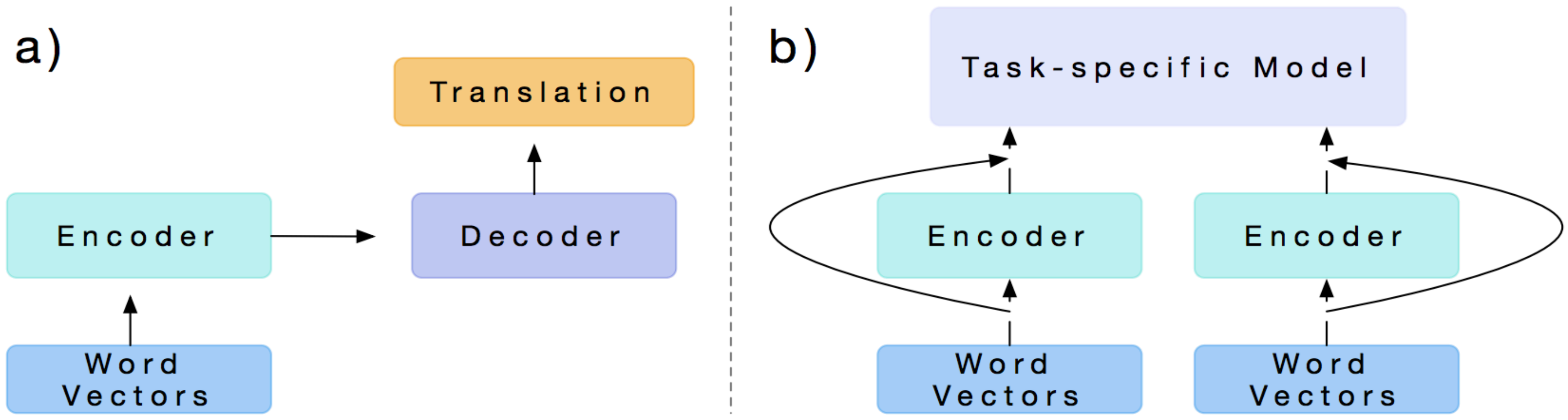


Figure 1: We a) train a two-layer, bidirectional LSTM as the encoder of an attentional sequence-to-sequence model for machine translation and b) use it to provide more context for other NLP models.

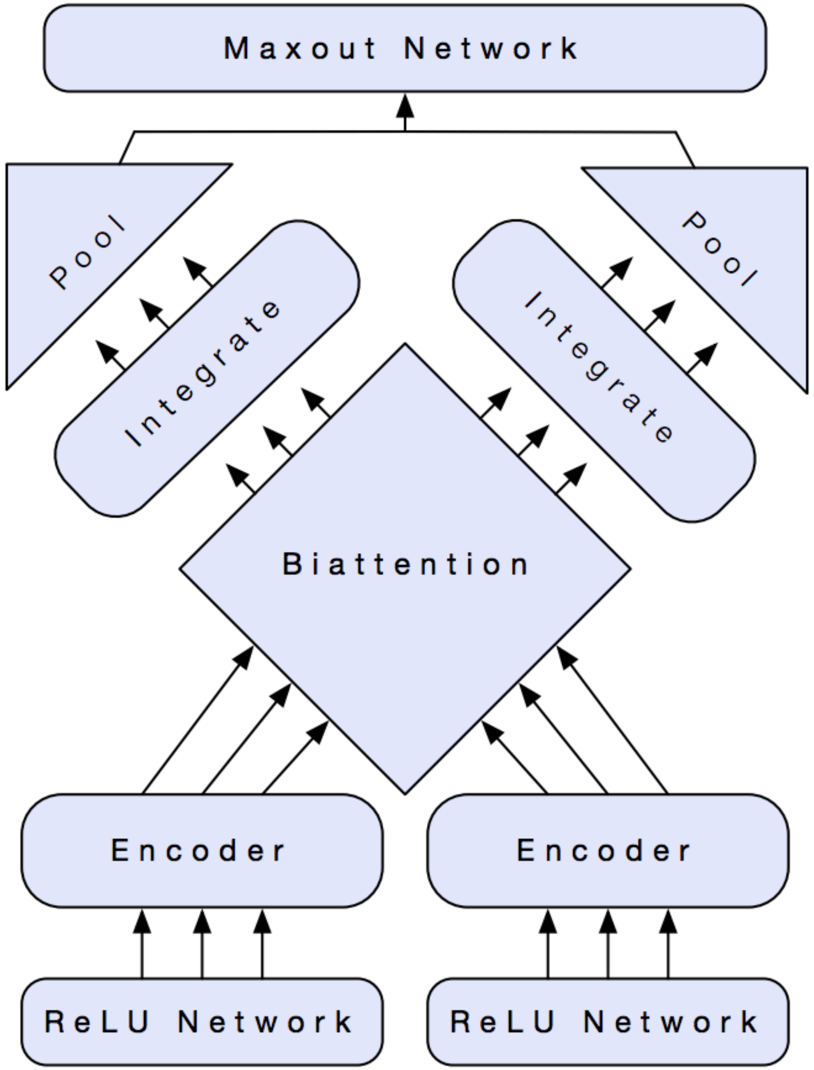
$$\mathrm{CoVe}(w) = \text{MT-LSTM}(\mathrm{GloVe}(w))$$

$$\tilde{w} = [\mathrm{GloVe}(w); \mathrm{CoVe}(w)]$$

Classification with CoVe

$$x = \text{biLSTM}\left(f(\tilde{w}^x)\right)$$

$$y = \text{biLSTM}\left(f(\tilde{w}^y)\right)$$



These sequences are each stacked along the time axis to get matrices X and Y.

$$A = XY^\top$$

$$A_x = \text{softmax}(A) \qquad A_y = \text{softmax}\left(A^\top\right)$$

$$C_x = A_x^\top X \qquad C_y = A_y^\top Y$$

$$X_{|y} = \text{biLSTM}\left([X; X - C_y; X \odot C_y]\right)$$

$$Y_{|x} = \text{biLSTM}\left([Y; Y - C_x; Y \odot C_x]\right)$$

$$\beta_x = \text{softmax}\left(X_{|y} v_1 + d_1\right) \qquad \beta_y = \text{softmax}\left(Y_{|x} v_2 + d_2\right)$$

$$x_{\text{self}} = X_{|y}^\top \beta_x \qquad y_{\text{self}} = Y_{|x}^\top \beta_y$$

$$x_{\text{pool}} = \left[\max(X_{|y}); \text{mean}(X_{|y}); \min(X_{|y}); x_{\text{self}}\right]$$

$$y_{\text{pool}} = \left[\max(Y_{|x}); \text{mean}(Y_{|x}); \min(Y_{|x}); y_{\text{self}}\right]$$

Question Answering with CoVe

$$x = \mathrm{biLSTM}\left(f(\tilde{w}^x)\right)$$

$$y = \mathrm{biLSTM}\left(f(\tilde{w}^y)\right)$$

f is a tanh activation

In this case, one of the sequences is the document and the other the question in the question-document pair.

Experiments

Dataset	Random	GloVe	GloVe+				
			Char	CoVe-S	CoVe-M	CoVe-L	Char+CoVe-L
SST-2	84.2	88.4	90.1	89.0	90.9	91.1	**91.2**
SST-5	48.6	53.5	52.2	54.0	54.7	54.5	**55.2**
IMDb	88.4	91.1	91.3	90.6	91.6	91.7	**92.1**
TREC-6	88.9	94.9	94.7	94.7	95.1	95.8	**95.8**
TREC-50	81.9	89.2	89.8	89.6	89.6	90.5	**91.2**
SNLI	82.3	87.7	87.7	87.3	87.5	87.9	**88.1**
SQuAD	65.4	76.0	78.1	76.5	77.1	79.5	**79.9**

Table 2: CoVe improves validation performance. CoVe has an advantage over character n-gram embeddings, but using both improves performance further. Models benefit most by using an MT-LSTM trained with MT-Large (CoVe-L). Accuracy reported for classification tasks; F1 for SQuAD.

Character embeddings: Character n-gram embeddings are trained by the same Skip-gram objective. We construct the character n-gram vocabulary in the training data and assign an embedding for each entry. The final character embedding is the average of the *unique* character n-gram embeddings of w_t. For example, the character n-grams ($n = 1, 2, 3$) of the word "Cat" are {C, a, t, #B#C, Ca, at, t#E#, #B#Ca, Cat, at#E#}, where "#B#" and "#E#" represent the beginning and the end of each word, respectively. Using the character embeddings efficiently provides morphological features. Each word is subsequently represented as x_t, the concatenation of its corresponding word and character embeddings shared across the tasks.[1]

Experiments

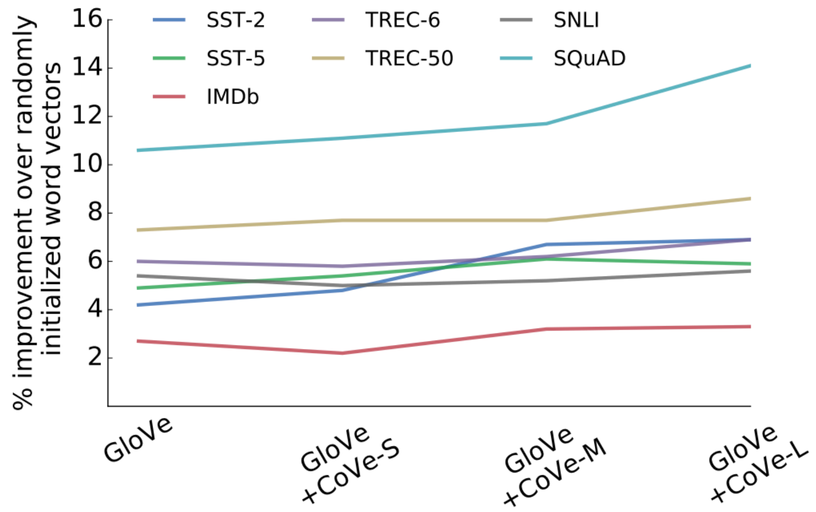
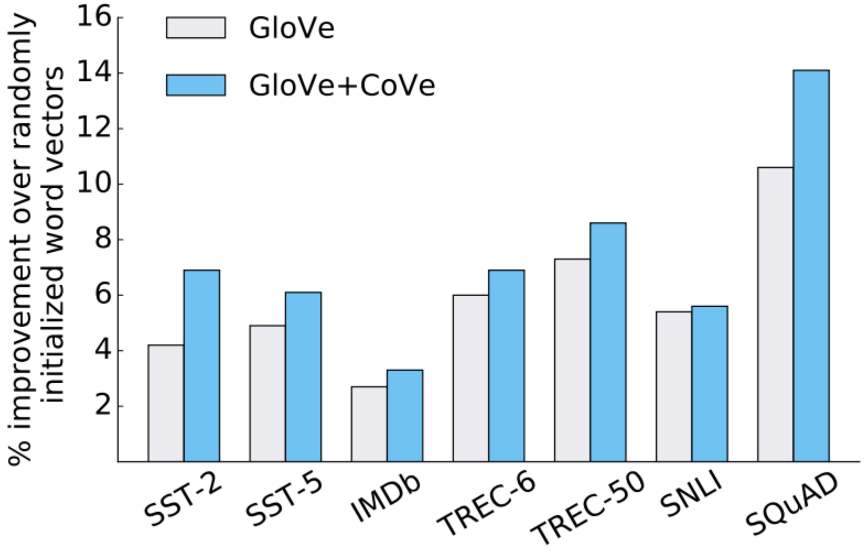
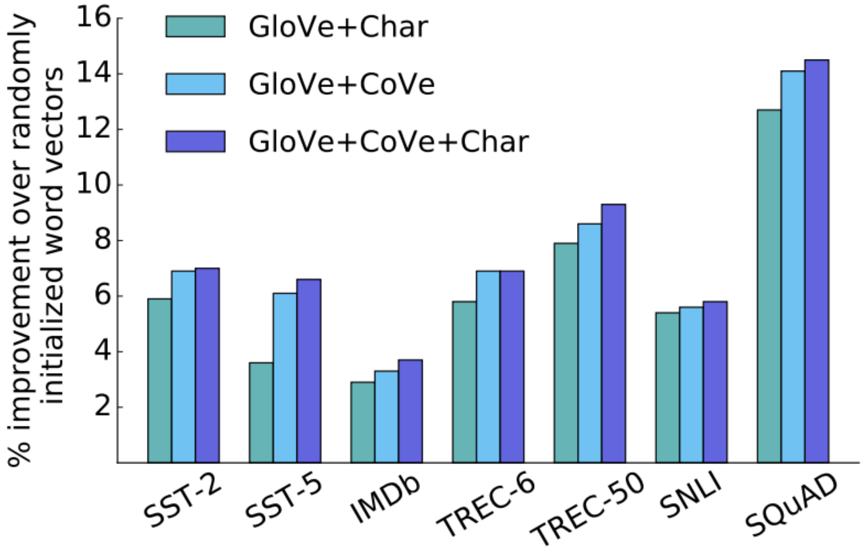


Figure 4: The Effects of MT Training Data



(a) CoVe and GloVe



(b) Cove and Characters

Experiments

	Model	Test		Model	Test
SST-2	P-LSTM [Wieting et al., 2016]	89.2	TREC-6	SVM [da Silva et al., 2011]	95.0
	CT-LSTM [Looks et al., 2017]	89.4		SVM [Van-Tu and Anh-Cuong, 2016]	95.2
	TE-LSTM [Huang et al., 2017]	89.6		DSCNN-P [Zhang et al., 2016]	95.6
	NSE [Munkhdalai and Yu, 2016a]	89.7		*BCN+Char+CoVe (Ours)*	*95.8*
	BCN+Char+CoVe (Ours)	*90.3*		TBCNN [Mou et al., 2015]	96.0
	bmLSTM [Radford et al., 2017]	**91.8**		**LSTM-CNN [Zhou et al., 2016]**	**96.1**
SST-5	MVN [Guo et al., 2017]	51.5	TREC-50	SVM [Loni et al., 2011]	89.0
	DMN [Kumar et al., 2016]	52.1		SNoW [Li and Roth, 2006]	89.3
	LSTM-CNN [Zhou et al., 2016]	52.4		*BCN+Char+CoVe (Ours)*	*90.2*
	TE-LSTM [Huang et al., 2017]	52.6		RulesUHC [da Silva et al., 2011]	90.8
	NTI [Munkhdalai and Yu, 2016b]	53.1		SVM [Van-Tu and Anh-Cuong, 2016]	91.6
	BCN+Char+CoVe (Ours)	***53.7***		**Rules [Madabushi and Lee, 2016]**	**97.2**
IMDb	*BCN+Char+CoVe (Ours)*	*91.8*	SNLI	DecAtt+Intra [Parikh et al., 2016]	86.8
	SA-LSTM [Dai and Le, 2015]	92.8		NTI [Munkhdalai and Yu, 2016b]	87.3
	bmLSTM [Radford et al., 2017]	92.9		re-read LSTM [Sha et al., 2016]	87.5
	TRNN [Dieng et al., 2016]	93.8		btree-LSTM [Paria et al., 2016]	87.6
	oh-LSTM [Johnson and Zhang, 2016]	94.1		600D ESIM [Chen et al., 2016]	88.0
	Virtual [Miyato et al., 2017]	**94.1**		***BCN+Char+CoVe (Ours)***	***88.1***

Table 3: Single model test accuracies for classification tasks.

Model	Reference	EM	F1
LR	Rajpurkar et al. [2016]	40.0	51.0
DCR	Yu et al. [2017]	62.5	72.1
M-LSTM+AP	Wang and Jiang [2017]	64.1	73.9
DCN+Char	Xiong et al. [2017]	65.4	75.6
BiDAF	Seo et al. [2017]	68.0	77.3
R-NET	Wang et al. [2017]	71.1	79.5
DCN+Char+CoVe	***Ours***	***71.3***	***79.9***

Table 4: Validation exact match and F1 for single-model question answering.

Pros:

- Contextualized word vectors

Cons:

- Only use the top layer of an LSTM
- Classification model lacks intuition
- Only experiments on classification and question answering tasks

Deep Contextualized Word Representations

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer

NAACL 2018

ELMo: Embeddings from Language Models

For each token t_k, a L-layer biLM computes a set of 2L + 1 representations

$$\begin{aligned} R_k &= \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ &= \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\}, \end{aligned}$$

where $\mathbf{h}_{k,0}^{LM}$ is the token layer and $\mathbf{h}_{k,j}^{LM} = [\overrightarrow{\mathbf{h}}_{k,j}^{LM}; \overleftarrow{\mathbf{h}}_{k,j}^{LM}]$, for each biLSTM layer.

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}$$

In (1), $\mathbf{s}^{task}$ are softmax-normalized weights and the scalar parameter γ^{task} allows the task model to scale the entire ELMo vector. γ is of practical importance to aid the optimization process

Using biLMs for supervised NLP tasks

To add ELMo to the supervised model, we first freeze the weights of the biLM and then concatenate the ELMo vector $\mathbf{ELMo}_k^{task}$ with $\mathbf{x}_k$ and pass the ELMo enhanced representation $[\mathbf{x}_k; \mathbf{ELMo}_k^{task}]$ into the task RNN.

For some tasks (e.g., SNLI, SQuAD), we observe further improvements by also including ELMo at the output of the task RNN by introducing another set of output specific linear weights and replacing $\mathbf{h}_k$ with $[\mathbf{h}_k; \mathbf{ELMo}_k^{task}]$.

Experiments

TASK	PREVIOUS SOTA		OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.

Experiments

- we found it beneficial to add a moderate amount of dropout to ELMo and in some cases to regularize the ELMo weights by adding $\lambda\|\mathbf{w}\|_2^2$ to the loss.

Task	Baseline	Last Only	All layers λ=1	All layers λ=0.001
SQuAD	80.8	84.7	85.0	**85.2**
SNLI	88.1	89.1	89.3	**89.5**
SRL	81.6	84.1	84.6	**84.8**

Table 2: Development set performance for SQuAD, SNLI and SRL comparing using all layers of the biLM (with different choices of regularization strength λ) to just the top layer.

Task	Input Only	Input & Output	Output Only
SQuAD	85.1	**85.6**	84.8
SNLI	88.9	**89.5**	88.7
SRL	**84.7**	84.3	80.9

Table 3: Development set performance for SQuAD, SNLI and SRL when including ELMo at different locations in the supervised model.

Experiments: What information is captured by the biLM's representations?

- Word sense disambiguation

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {. . . }	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {. . . }	{. . . } they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Table 4: Nearest neighbors to "play" using GloVe and the context embeddings from a biLM.

Experiments: What information is captured by the biLM's representations?

- Word sense disambiguation
- Basic syntax

  accuracies using the first biLM layer are higher than the top layer

  different layers in the biLM represent different types of information

Model	F_1
WordNet 1st Sense Baseline	65.9
Raganato et al. (2017a)	69.9
Iacobacci et al. (2016)	**70.1**
CoVe, First Layer	59.4
CoVe, Second Layer	*64.7*
biLM, First layer	67.4
biLM, Second layer	*69.0*

Table 5: All-words fine grained WSD F_1. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

Model	Acc.
Collobert et al. (2011)	97.3
Ma and Hovy (2016)	97.6
Ling et al. (2015)	**97.8**
CoVe, First Layer	*93.3*
CoVe, Second Layer	92.8
biLM, First Layer	*97.3*
biLM, Second Layer	96.8

Table 6: Test set POS tagging accuracies for PTB. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

Experiments: Sample Effciency

- the SRL model reaches a maximum development F1 after 486 epochs of training without ELMo. After adding ELMo, the model exceeds the baseline maximum at epoch 10.

- In addition, ELMo-enhanced models use smaller training sets more efficiently than models without ELMo.

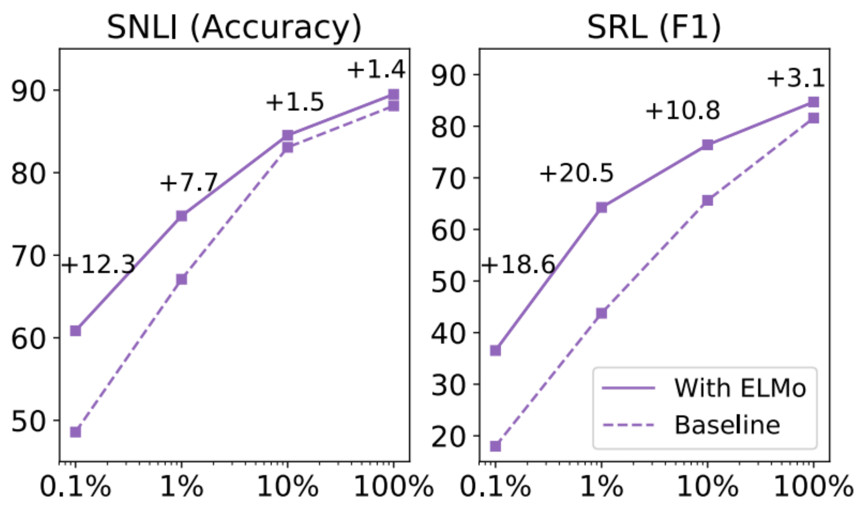


Figure 1: Comparison of baseline vs. ELMo performance for SNLI and SRL as the training set size is varied from 0.1% to 100%.

Experiments: Visualization of learned weights

- At the input layer, the task model favors the first biLSTM layer. For coreference and SQuAD, the this is strongly favored, but the distribution is less peaked for the other tasks. The output layer weights are relatively balanced, with a slight preference for the lower layers.

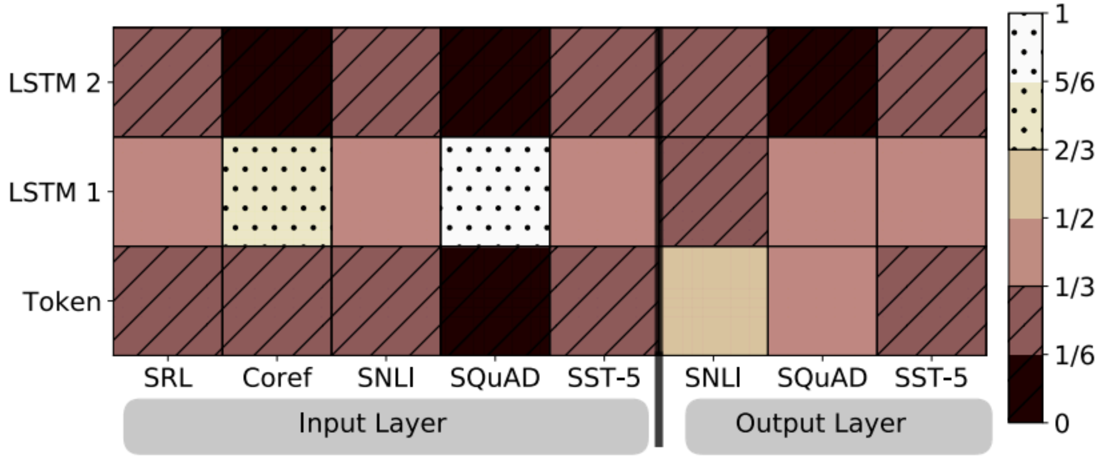


Figure 2: Visualization of softmax normalized biLM layer weights across tasks and ELMo locations. Normalized weights less then $1/3$ are hatched with horizontal lines and those greater then $2/3$ are speckled.

Pros:

Contextualized word vectors

Good amount of experiments

Integrate multiple layers of LSTMs

Show that the higher-level LSTM states capture context-dependent aspects of word meaning while lower-level states model aspects of syntax

Cons:

Linear combination

Inspiration:

Deep layer aggregation [Fisher et al., 2018] for Transformer