Using monolingual data

Review: the noisy channel

Bayes' rule

$$p(\mathbf{e} \mid \mathbf{f}) = \frac{p_{LM}(\mathbf{e}) \times p_{TM}(\mathbf{f} \mid \mathbf{e})}{p(\mathbf{f})}$$

Review: the noisy channel

chain rule $$p(\mathbf{e}, \mathbf{f}) = p_{LM}(\mathbf{e}) \times p_{TM}(\mathbf{f} \mid \mathbf{e})$$

Review: the noisy channel

chain rule

$$p(\mathbf{e}, \mathbf{f}) = p_{LM}(\mathbf{e}) \times p_{TM}(\mathbf{f} \mid \mathbf{e})$$

language model

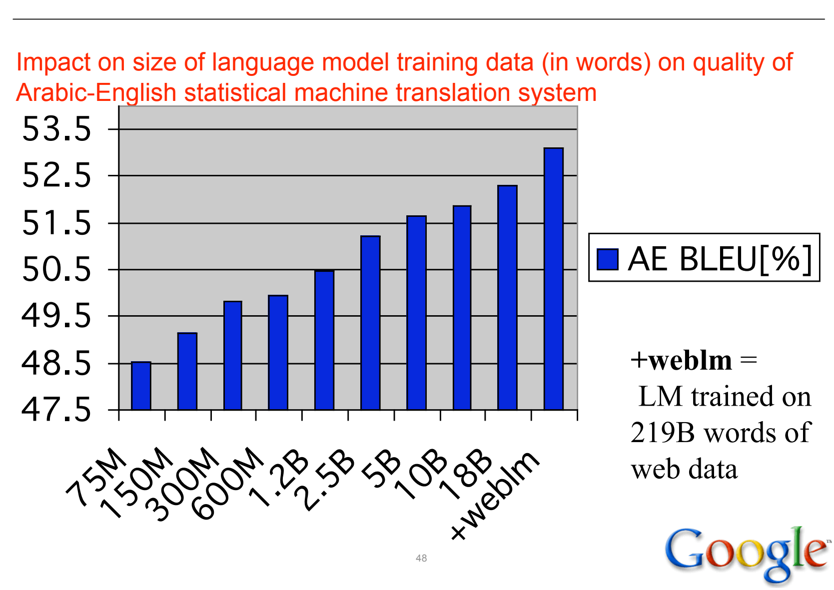
translation model (conditional language model)

How much training data for each of these?

Review: the noisy channel

chain rule $$p(\mathbf{e}, \mathbf{f}) = p_{LM}(\mathbf{e}) \times p_{TM}(\mathbf{f} \mid \mathbf{e})$$

language model



translation model (conditional language model)

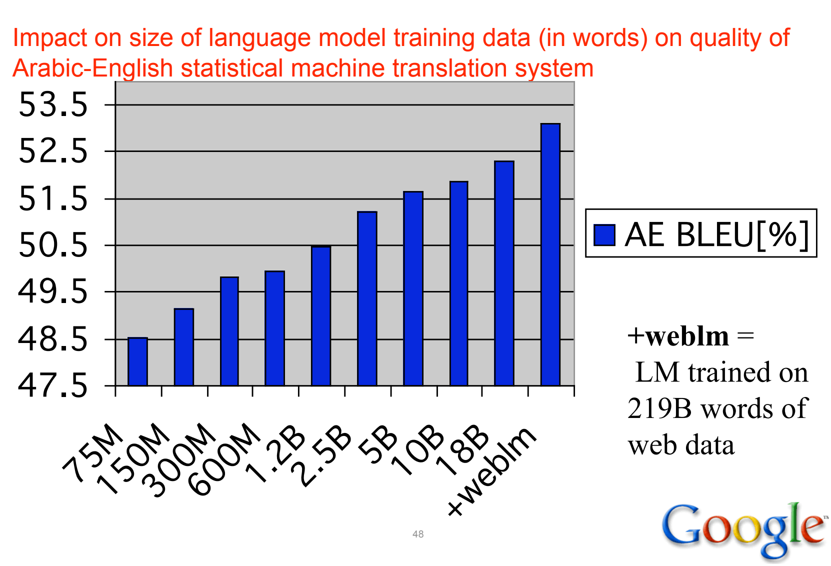
maybe 1B words

Review: neural MT

Er, there's no decomposition here.

$$p(\mathbf{e} \mid \mathbf{f}) = p(\mathbf{e} \mid \mathbf{f})$$

language model



translation model (conditional language model)

maybe 1B words

Where did this go?

Is this data still useful?

Attempt 1

Gulcehre et al., June 2015

- MaxEnt (NN mumbo-jumbo term: “shallow fusion”)

$$\log p(\mathbf{e} \mid \mathbf{f}) = \log p_{TM}(\mathbf{e} \mid \mathbf{f}) + \beta \log p_{LM}(\mathbf{e})$$

Attempt 1

Gulcehre et al., June 2015

- MaxEnt (NN mumbo-jumbo: “shallow fusion”)

$$\log p(\mathbf{e} \mid \mathbf{f}) = \log p_{TM}(\mathbf{e} \mid \mathbf{f}) + \beta \log p_{LM}(\mathbf{e})$$

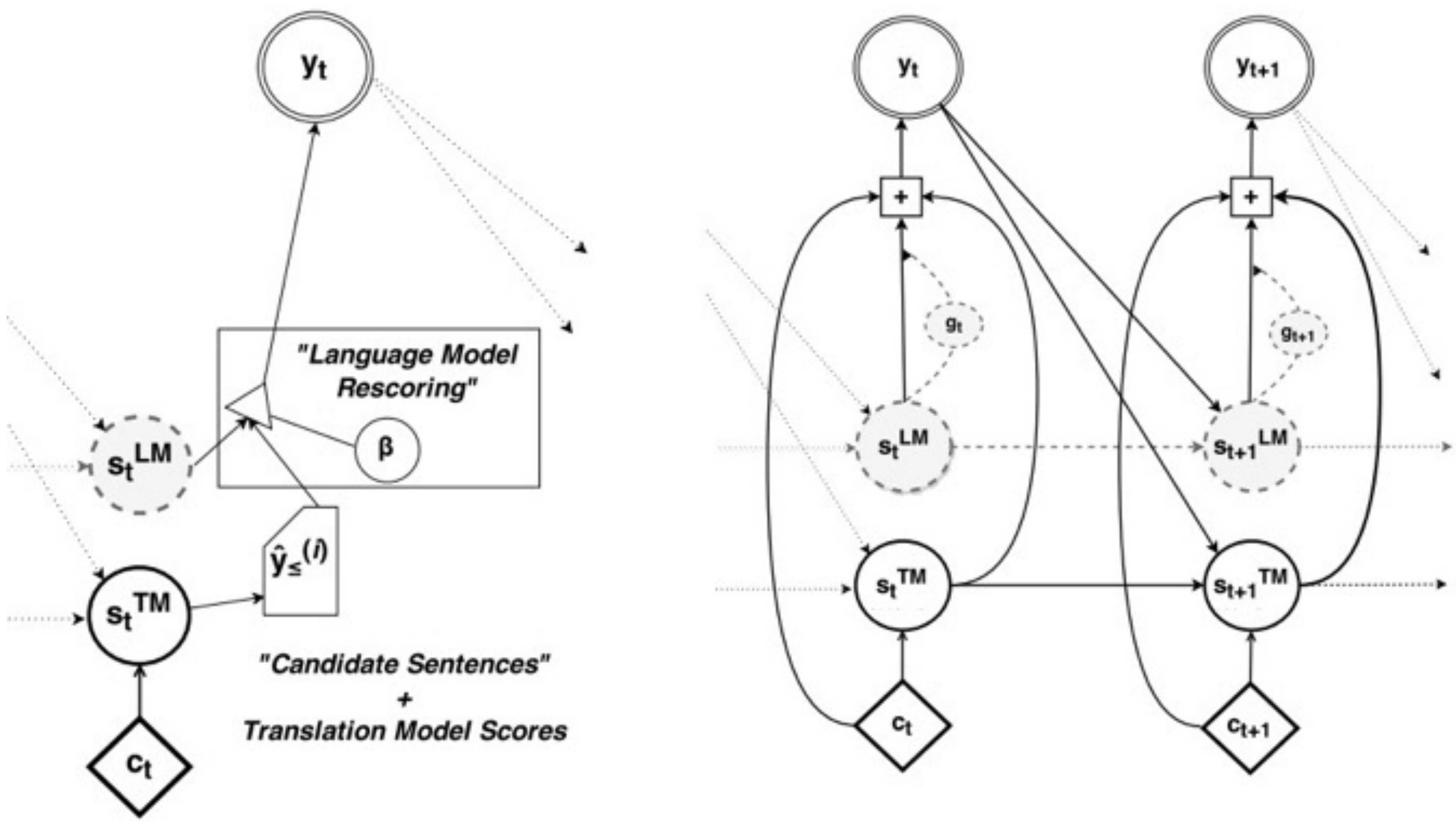
Tune this parameter on development set, once other models are learned

Note difference from Bayes’ rule!

Attempt 1

Gulcehre et al., June 2015

- “Deep fusion”: output distribution a function of (fixed) LM hidden state and (learned) TM hidden state



(a) Shallow Fusion (Sec. 4.1)

(b) Deep Fusion (Sec. 4.2)

Attempt 1

Gulcehre et al., June 2015

- “Deep fusion”: output distribution a function of (fixed) LM hidden state and (learned) TM hidden state

	De-En		Cs-En	
	Dev	Test	Dev	Test
NMT Baseline	25.51	23.61	21.47	21.89
Shallow Fusion	25.53	23.69	21.95	22.18
Deep Fusion	**25.88**	**24.00**	**22.49**	**22.36**

	Development Set		Test Set			
	dev2010	**tst2010**	**tst2011**	**tst2012**	**tst2013**	**Test 2014**
Previous Best (Single)	15.33	17.14	18.77	18.62	18.88	-
Previous Best (Combination)	-	17.34	18.83	18.93	18.70	-
NMT	14.50	18.01	18.40	18.77	19.86	18.64
NMT+LM (Shallow)	14.44	17.99	18.48	18.80	19.87	18.66
NMT+LM (Deep)	**15.69**	**19.34**	**20.17**	**20.23**	**21.34**	**20.56**

Attempt 2

Sennrich et al., Nov 2015

- Idea 1: Train encoder-decoder on parallel and monolingual target-language data.
    - On monolingual data, set context to 0 (essentially, train decoder without encoder).

Attempt 2

Sennrich et al., Nov 2015

- Idea 2: Very old idea: backtranslation
  - MT folklore: in English->Russian->English: “The spirit is willing, but the flesh is weak” -> “The vodka is good, but the meat is rotten”
- Basic idea: train two systems, backtranslate English to French, then retrain English French-English system on resulting data.

Attempt 2

Sennrich et al., Nov 2015

- Monolingual: Empty context vector
- Synthetic: Backtranslation

name	training instances	BLEU newstest2014 single	BLEU newstest2014 ens-4	BLEU newstest2015 single	BLEU newstest2015 ens-4
syntax-based (Sennrich and Haddow, 2015)		22.6	-	24.4	-
Neural MT (Jean et al., 2015b)		-	-	22.4	-
parallel	37m (parallel)	19.9	20.4	22.8	23.6
+monolingual	49m (parallel) / 49m (monolingual)	20.4	21.4	23.2	24.6
+synthetic	44m (parallel) / 36m (synthetic)	**22.7**	**23.8**	**25.7**	**26.5**

Table 3: English→German translation performance (BLEU) on WMT training/test sets. Ens-4: ensemble of 4 models. Number of training instances varies due to differences in training time and speed.

Attempt 2

Sennrich et al., Nov 2015

- Synthetic: Backtranslation

	BLEU	
name	2014	2015
PBSMT (Haddow et al., 2015)	28.8	29.3
NMT (Gülçehre et al., 2015)	23.6	-
+shallow fusion	23.7	-
+deep fusion	24.0	-
parallel	25.9	26.7
+synthetic	**29.5**	**30.4**
+synthetic (ensemble of 4)	30.8	31.6

Table 5: German→English translation performance (BLEU) on WMT training/test sets (newstest2014; newstest2015).

Attempt 2

Sennrich et al., Nov 2015

- Synthetic: Backtranslation

phrase-based MT {

system	BLEU WMT	BLEU IWSLT
parallel	20.1	21.5
+synthetic	20.8	21.6
PBSMT gain	+0.7	+0.1
NMT gain	+2.9	+1.2

Table 8: Phrase-based SMT results (English→German) on WMT test sets (average of newstest201{4,5}), and IWSLT test sets (average of tst201{3,4,5}), and average BLEU gain from adding synthetic data for both PBSMT and NMT.

Attempt 2

Sennrich et al., Nov 2015

- Synthetic: Backtranslation

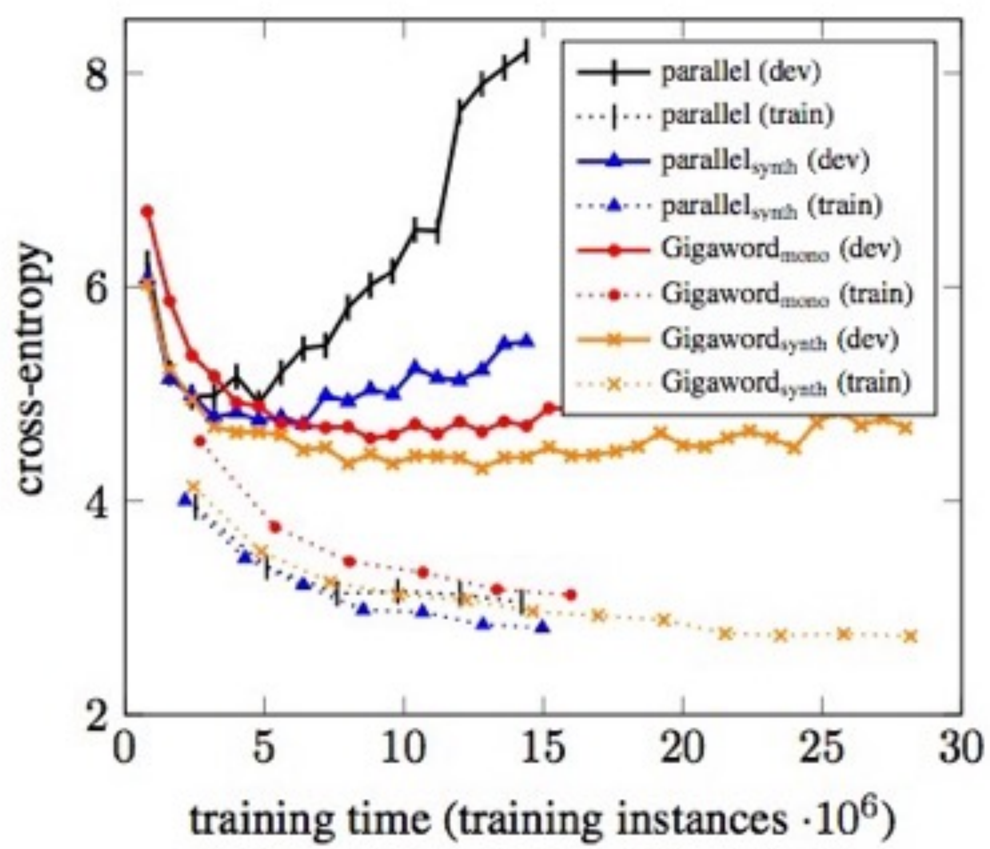


Figure 1: Turkish→English training and development set (tst2010) cross-entropy as a function of training time (number of training instances) for different systems.

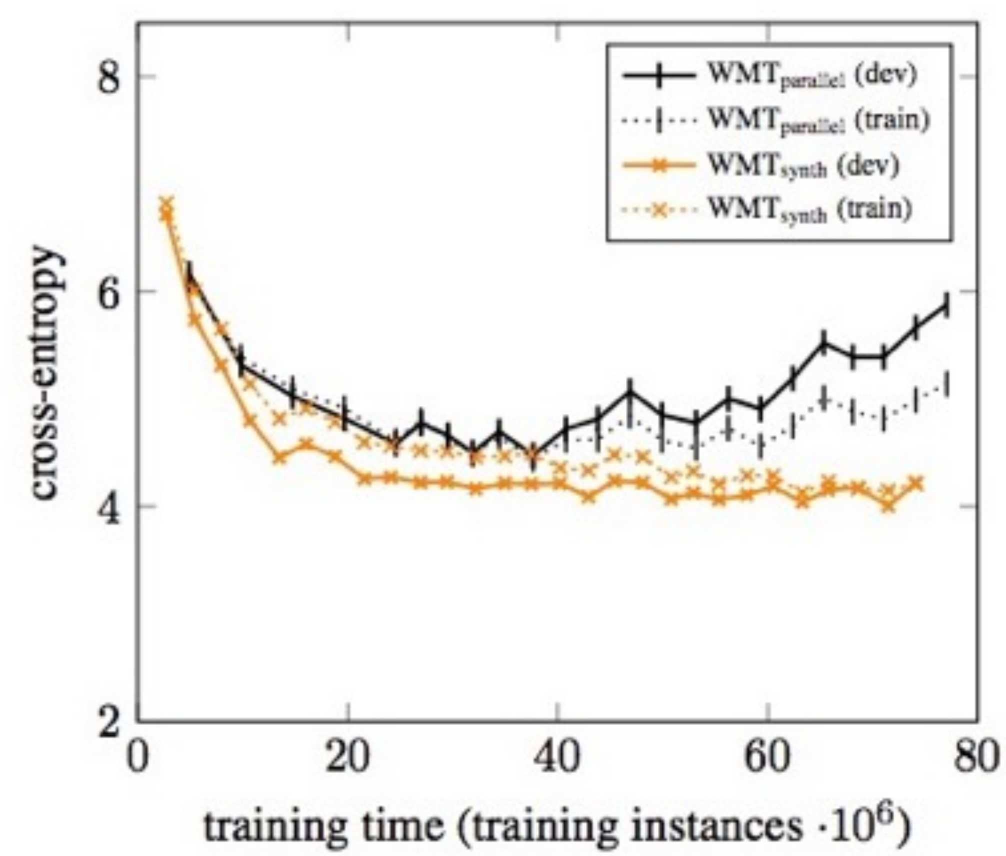


Figure 2: English→German training and development set (newstest2013) cross-entropy as a function of training time (number of training instances) for different systems.

Summary: LMs in NMT

- Monday’s discussion:
  - Phrase-based MT may be better for **adequacy**.
  - Neural MT may be better for **fluency**.
- Do we need a LM if our MT is already fluent?
- More generally: if our discriminative model has enough capacity, do we need a strong prior?
- On the other hand: denoising the input seems to work!

Hybrid systems

- Monday's discussion:
  - Phrase-based MT may be better for **adequacy**.
  - Neural MT may be better for **fluency**.
- How can we combine these benefits?

Simpler: LMs

Q: How would we combine a neural and an n-gram language model?

Simpler: LMs

Q: How would we combine a neural and an n-gram language model?

interpolation of *k* LMs:

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

*k*th weight
(weights must sum to 1)

*k*th LM

Simpler: LMs

Neubig & Dyer 2016

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

interpolation of *k* LMs as matrix multiplication:

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^\intercal} = \underbrace{\begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,K} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,K} \\ \vdots & \vdots & \ddots & \vdots \\ d_{J,1} & d_{J,2} & \cdots & d_{J,K} \end{bmatrix}}_{\text{Distribution matrix } D} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_K \end{bmatrix}}^{\text{Coefficients } \boldsymbol{\lambda}^\intercal}$$

*k*th weight (weights must sum to 1)

*k*th LM

Simpler: LMs

Neubig & Dyer 2016

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

interpolation of *k* LMs as matrix multiplication:

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^{\mathsf{T}}} = \underbrace{\begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,K} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,K} \\ \vdots & \vdots & \ddots & \vdots \\ d_{J,1} & d_{J,2} & \cdots & d_{J,K} \end{bmatrix}}_{\text{Distribution matrix } D} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_K \end{bmatrix}}^{\text{Coefficients } \boldsymbol{\lambda}^{\mathsf{T}}}$$

probability of *i*th element

Simpler: LMs

Neubig & Dyer 2016

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

heuristic **interpolation** of LMs (e.g. Kneser-Ney)

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^\top} = \underbrace{\begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,N} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ d_{J,1} & d_{J,2} & \cdots & d_{J,N} \end{bmatrix}}_{\text{Count-based probabilities } P_C(w_i = j|w_{i-n+1}^{i-1})} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix}}^{\text{Heuristic interp. coefficients } \boldsymbol{\lambda}^\top}$$

Simpler: LMs

Neubig & Dyer 2016

$$\underbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}_{\text{Probabilities } \boldsymbol{p}^\intercal} = \underbrace{\begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,N} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ d_{J,1} & d_{J,2} & \cdots & d_{J,N} \end{bmatrix}}_{\text{Count-based probabilities } P_C(w_i = j | w_{i-n+1}^{i-1})} \underbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix}}_{\text{Heuristic interp. coefficients } \boldsymbol{\lambda}^\intercal}$$

Do we know any other way to produce this vector?

Simpler: LMs

Neubig & Dyer 2016

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^\intercal} = \underbrace{\begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,N} \\ d_{1,2} & d_{2,2} & \cdots & d_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ d_{J,1} & d_{J,2} & \cdots & d_{J,N} \end{bmatrix}}_{\text{Count-based probabilities } P_C(w_i = j | w_{i-n+1}^{i-1})} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix}}^{\text{Result of softmax(NN}(\boldsymbol{c}))}$$

Simpler: LMs

Neubig & Dyer 2016

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

matrix **interpretation** of RNNLM

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^\top} = \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{\text{J-by-J identity matrix } I} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_J \end{bmatrix}}^{\text{Result of softmax(NN}(\boldsymbol{c}))}$$

Simpler: LMs

Neubig & Dyer 2016

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

matrix **interpretation** of RNNLM

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^{\mathsf{T}}} = \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{\text{J-by-J identity matrix } I} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_J \end{bmatrix}}^{\text{Result of softmax(NN}(\boldsymbol{c}))}$$

Simpler: LMs

Neubig & Dyer 2016

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

neural interpolation of count-based and RNNLM

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^\intercal} = \underbrace{\begin{bmatrix} d_{1,1} & \cdots & d_{1,N} & 1 & \cdots & 0 \\ d_{2,1} & \cdots & d_{2,N} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ d_{J,1} & \cdots & d_{J,N} & 0 & \cdots & 1 \end{bmatrix}}_{\text{Count-based probabilities } \underline{\text{and}}\ J\text{-by-}J \text{ identity matrix}} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{J+N} \end{bmatrix}}^{\text{Result of softmax}(\text{NN}(\boldsymbol{c}))}$$

Simpler: LMs

Neubig & Dyer 2016

$$P(w_i|\boldsymbol{c}) = \sum_{k=1}^{K} \lambda_k(\boldsymbol{c}) P_k(w_i|\boldsymbol{c})$$

matrix **interpretation** of RNNLM

$$\overbrace{\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix}}^{\text{Probabilities } \boldsymbol{p}^\intercal} = \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{\text{J-by-J identity matrix } I} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_J \end{bmatrix}}^{\text{Result of softmax(NN}(\boldsymbol{c}))}$$

Simpler: LMs

Neubig & Dyer 2016

- Training: block dropout (force count-based interpolation weights to zero for first several epochs)

	Dist.	Interp.	PPL
(1)	KN	HEUR	140.8/156.5
(2)	δ	LSTM	105.9/116.9
(3)	KN	LSTM	135.2/149.1
(4)	KN,δ	LSTM -BlDO	108.4/130.4
(5)	KN,δ	LSTM +BlDO	95.3 /104.5

Simpler: LMs

Neubig & Dyer 2016

- Training: block dropout (force count-based interpolation weights to zero for first several epochs).
- Other observations: seem to help with low-frequency words (where count-based LMs tend to be better)

Applied to MT

Arthur et al. 2016

- Observation: phrase-based MT better at adequacy (often due to handling of rare words).

Input:	I come from Tunisia.
Reference:	チュニジアの 出身です。 Chunisia no shusshindesu. *(I'm from Tunisia.)*
System:	ノルウェーの 出身です。 Noruue- no shusshindesu. *(I'm from Norway.)*

Figure 1: An example of a mistake made by NMT on low-frequency content words.

Applied to MT

Arthur et al. 2016

- Observation: phrase-based MT better at adequacy (often due to handling of rare words).

Input:	I come from Tunisia.
Reference:	チュニジア の 出身です。
	Chunisia no shusshindesu.
	(I'm from Tunisia.)
System:	ノルウェー の 出身です。
	Noruue- no shusshindesu.
	(I'm from Norway.)

Figure 1: An example of a mistake made by NMT on low-frequency content words.

Applied to MT

Arthur et al. 2016

- Observation: phrase-based MT better at adequacy (often due to handling of rare words).
- Basic idea: interpolate conditional neural LM with IBM Model 1 probabilities.

$$p_o(e_i|F, e_1^{i-1}) =$$

$$\begin{bmatrix} p_l(e_i = 1|F, e_1^{i-1}) & p_m(e = 1|F, e_1^{i-1}) \\ \vdots & \vdots \\ p_l(e_i = |V_e||F, e_1^{i-1}) & p_m(e = |V_e||F, e_1^{i-1}) \end{bmatrix} \begin{bmatrix} \lambda \\ 1 - \lambda \end{bmatrix}$$

Applied to MT

Arthur et al. 2016

- Observation: phrase-based MT better at adequacy (often due to handling of rare words).
- Basic idea: interpolate conditional neural LM with IBM Model 1 probabilities.

System	BTEC			KFTT		
	BLEU	NIST	RECALL	BLEU	NIST	RECALL
`pbmt`	48.18	6.05	27.03	22.62	5.79	13.88
`hiero`	52.27	6.34	24.32	22.54	5.82	12.83
`attn`	48.31	5.98	17.39	20.86	5.15	17.68
`auto-bias`	**49.74***	**6.11***	**50.00**	**23.20**†	**5.59**†	**19.32**
`hyb-bias`	**50.34**†	**6.10***	**41.67**	**22.80**†	**5.55**†	16.67

Where to next?

- Coursework 2 due Monday (right before lecture).
- You've seen many things that are likely to be useful to you in Coursework 3.
- Next two weeks: more advanced ideas
  Including modeling more language phenomena
- Week 10: final lectures (won't need to understand these for coursework).