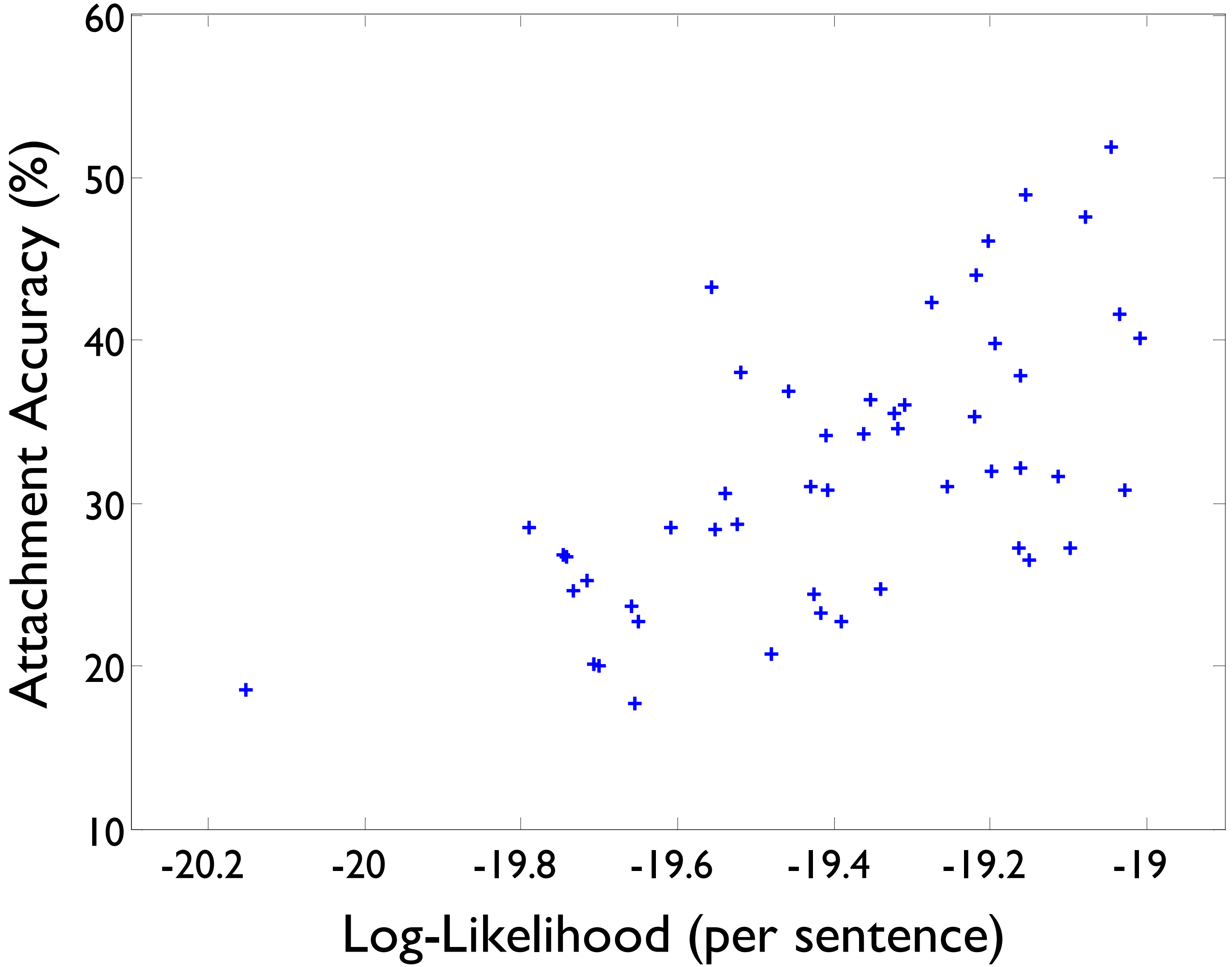
Concavity and Initialization for Unsupervised Dependency Parsing

Kevin Gimpel Noah A. Smith

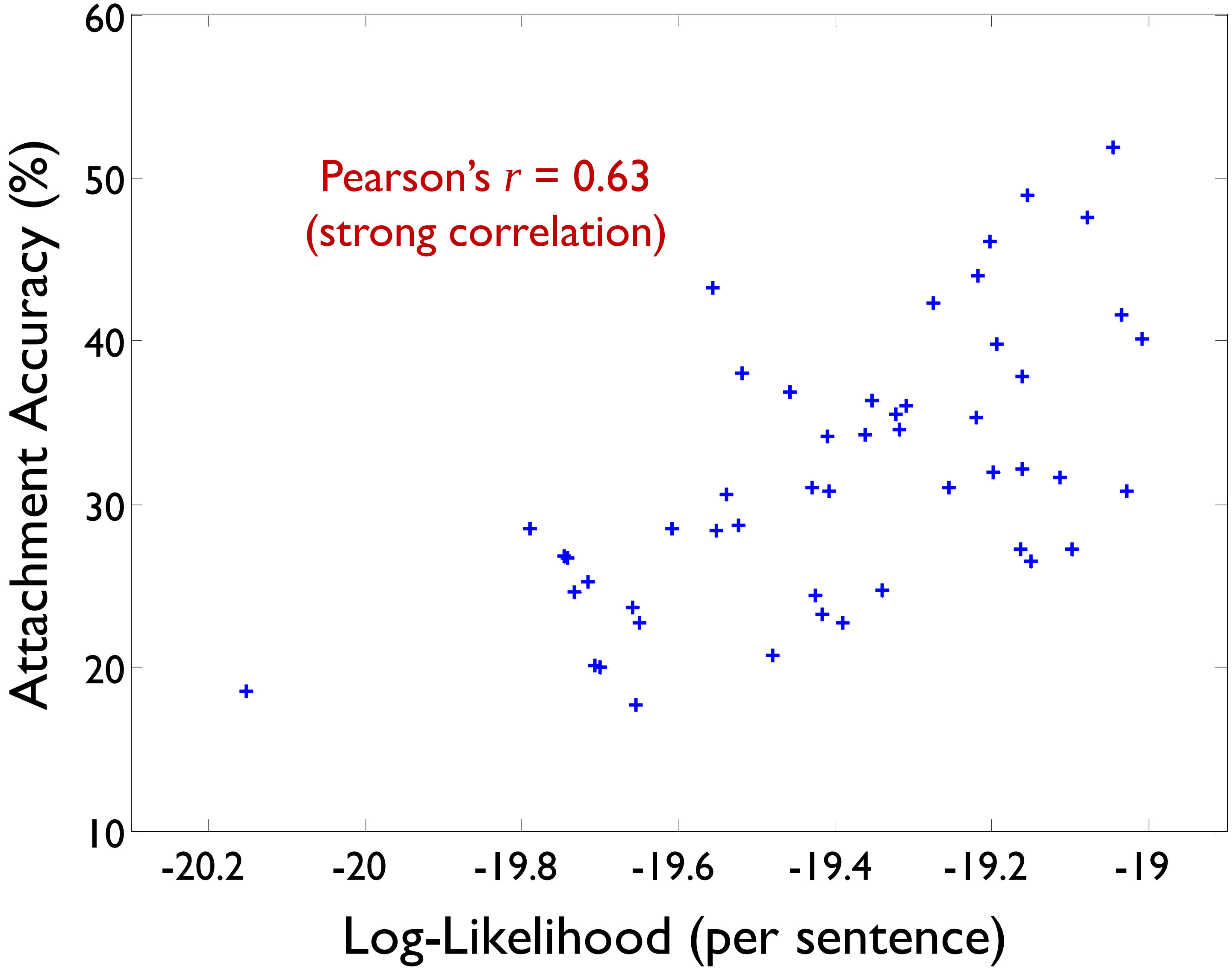
Unsupervised learning in NLP $\xrightarrow{\text{(typically)}}$ non-convex optimization

Dependency Model with Valence (Klein & Manning, 2004)

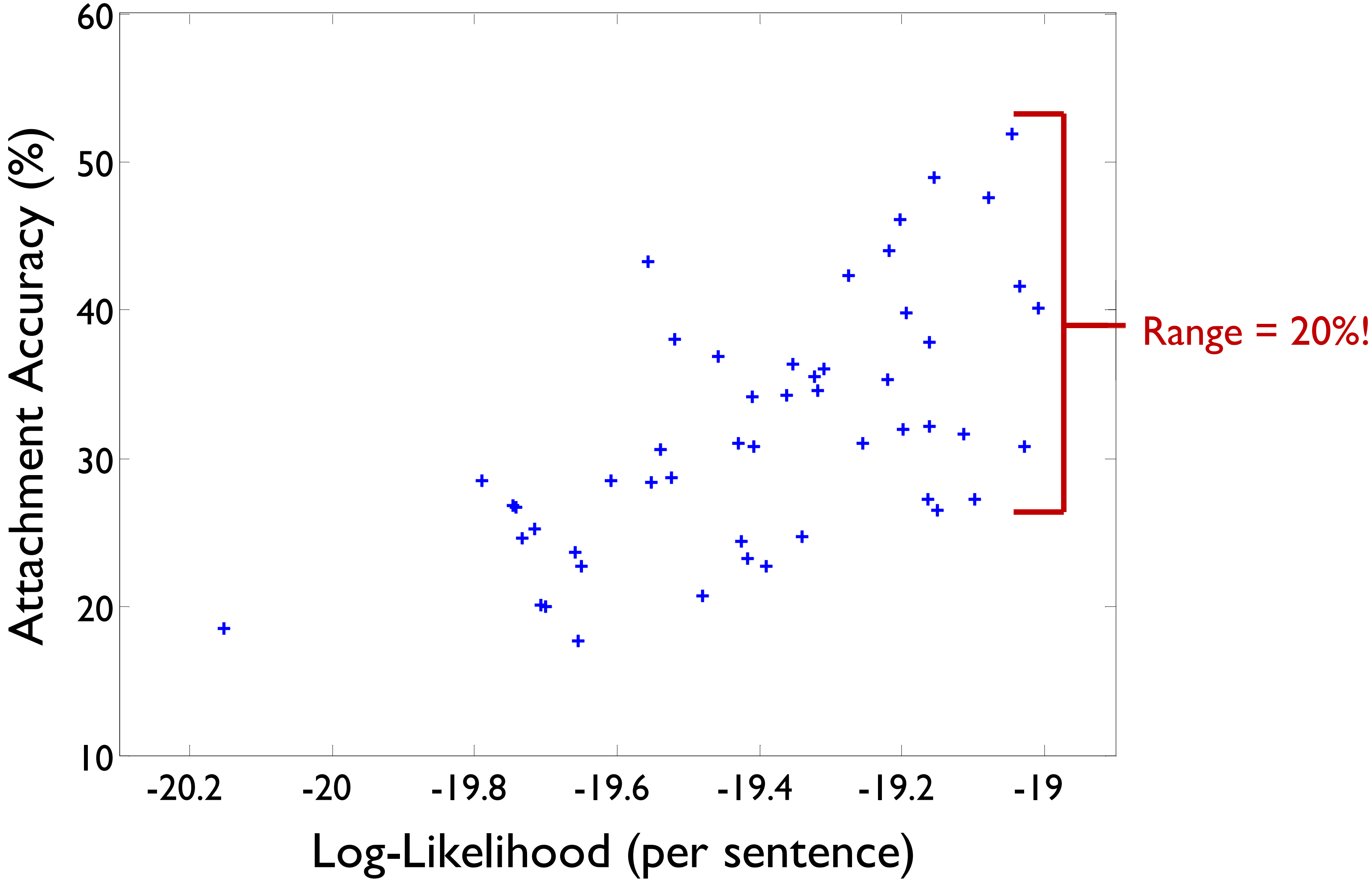


EM with 50 Random Initializers

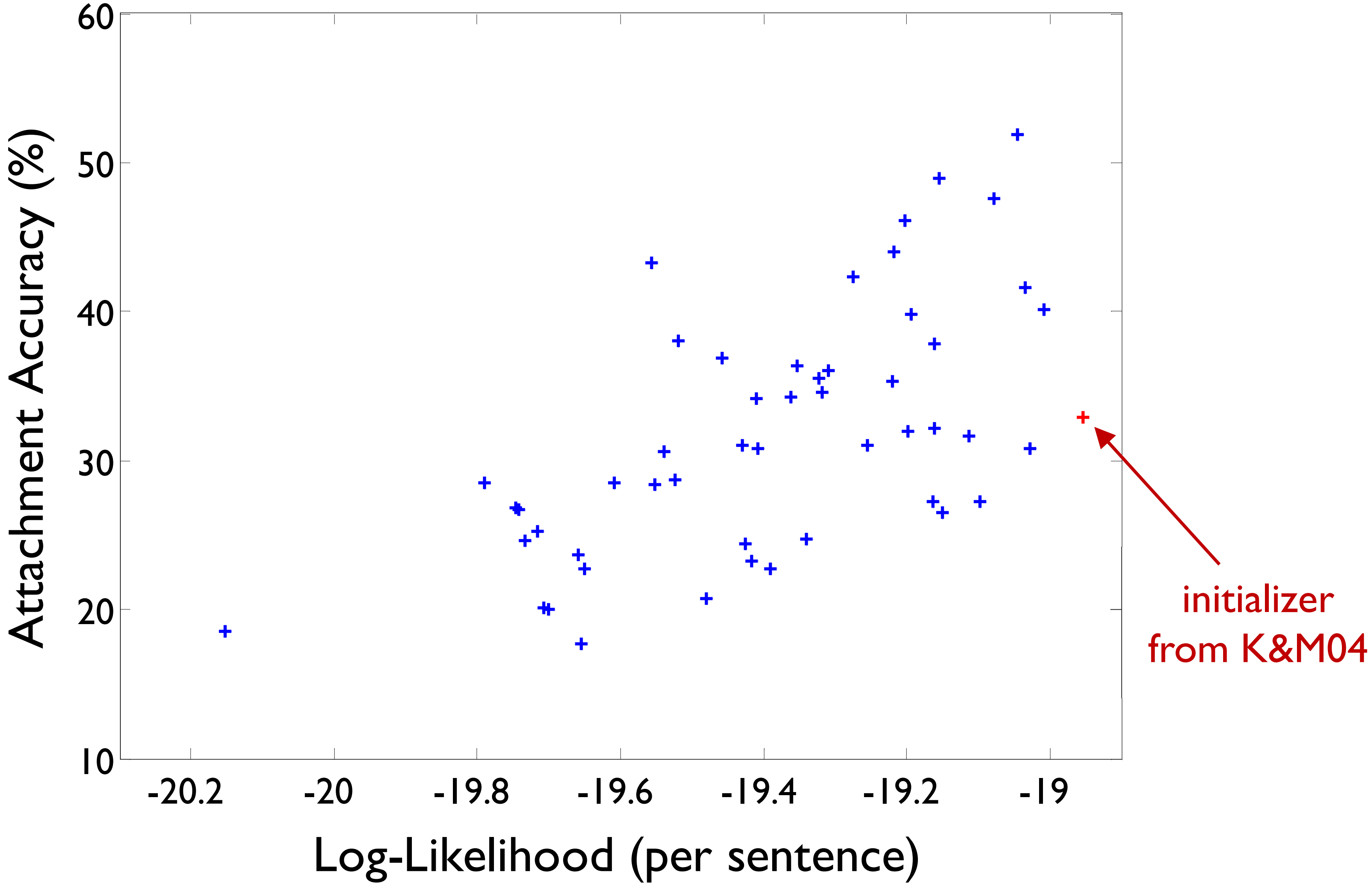
Dependency Model with Valence (Klein & Manning, 2004)



Dependency Model with Valence (Klein & Manning, 2004)



Dependency Model with Valence (Klein & Manning, 2004)



How has this been addressed?

- Scaffolding / staged training (Brown et al., 1993; Elman, 1993; Spitkovsky et al., 2010)
- Curriculum learning (Bengio et al., 2009)
- Deterministic annealing (Smith & Eisner, 2004), Structural annealing (Smith & Eisner, 2006)
- Continuation methods (Allgower & Georg, 1990)

Example: Word Alignment

IBM Model 1 — Brown et al. (1993)

↓

HMM Model

↓

IBM Model 4

Example: Word Alignment

CONCAVE

IBM Model 1 — Brown et al. (1993)

↓

HMM Model

↓

IBM Model 4

Unsupervised learning in NLP $\xrightarrow{\text{(typically)}}$ non-convex optimization

$$\text{Unsupervised learning in NLP} \xrightarrow{\text{(typically)}} \text{non-convex optimization}$$

Except IBM Model 1 for word alignment

(which has a concave log-likelihood function)

IBM Model 1 (Brown et al., 1993)

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \frac{1}{|f|+1} \; t(e_j \mid f_i)$$

IBM Model 1 (Brown et al., 1993)

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \underbrace{\frac{1}{|f|+1}}_{\text{alignment probability}} \underbrace{t(e_j \mid f_i)}_{\text{translation probability}}$$

IBM Model 1 (Brown et al., 1993)

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \underbrace{\frac{1}{|f|+1}}_{\text{alignment probability}} \underbrace{t(e_j \mid f_i)}_{\text{translation probability}}$$

IBM Model 2

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} a(i \mid j, |f|, |e|)\, t(e_j \mid f_i)$$

ARK lti Carnegie Mellon

IBM Model 1 $\boxed{\textbf{CONCAVE}}$

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \underbrace{\frac{1}{|f|+1}}_{\text{alignment probability}} \underbrace{t(e_j \mid f_i)}_{\text{translation probability}}$$

IBM Model 2 $\boxed{\textbf{NOT CONCAVE}}$

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} a(i \mid j, |f|, |e|)\, t(e_j \mid f_i)$$

IBM Model 1 $\boxed{\textbf{CONCAVE}}$

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \underbrace{\frac{1}{|f|+1}}_{\text{alignment probability}} \underbrace{t(e_j \mid f_i)}_{\text{translation probability}}$$

IBM Model 2 $\boxed{\textbf{NOT CONCAVE}}$

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \underbrace{a(i \mid j, |f|, |e|)\, t(e_j \mid f_i)}_{\text{product of parameters within log-sum}}$$

ARK lti Carnegie Mellon

IBM Model 1 $\boxed{\textbf{CONCAVE}}$

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \underbrace{\frac{1}{|f|+1}} \underbrace{t(e_j \mid f_i)}$$

For concavity:

1 parameter is permitted for each atomic piece of latent structure.

No atomic piece of latent structure can affect any other piece.

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0}^{|f|} \underbrace{a(i \mid j, |f|, |e|)\, t(e_j \mid f_i)}_{\text{product of parameters within log-sum}}$$

Unsupervised learning in NLP $\xrightarrow{\text{(typically)}}$ non-convex optimization

Except IBM Model 1 for word alignment

(which has a concave log-likelihood function)

What models can we build without sacrificing concavity?

For concavity:

1 parameter is permitted for each atomic piece of latent structure.

No atomic piece of latent structure can affect any other piece.

For concavity:

1 parameter is permitted for each atomic piece of latent structure.

No $\underbrace{\text{atomic piece of latent structure}}_{\text{single dependency arc}}$ can affect any other piece.

For concavity:

1 parameter is permitted for each atomic piece of latent structure.

No $\underbrace{\text{atomic piece of latent structure}}_{\text{single dependency arc}}$ can affect any other piece.

Every dependency arc must be independent,
so **we can't use a tree constraint**

For concavity:

1 parameter is permitted for each atomic piece of latent structure.

No $\underbrace{\text{atomic piece of latent structure}}_{\text{single dependency arc}}$ can affect any other piece.

Every dependency arc must be independent,
so **we can’t use a tree constraint**

Only one parameter allowed per dependency arc

For concavity:

1 parameter is permitted for each atomic piece of latent structure.

No atomic piece of latent structure can affect any other piece.

single dependency arc

Our Model:

Like IBM Model 1, but we generate the same sentence again, aligning words to the original sentence (cf. Brody, 2010)

$$\log p(\boldsymbol{e} \mid \boldsymbol{e}') = \log \epsilon + \sum_{j=1}^{|e|} \log \sum_{i=0, i \neq j}^{|e|} \frac{1}{|e|+1} \; \mathit{child}(e_j \mid e'_i)$$

$ Vikings came in longboats from Scandinavia in 1000 AD

ARK lti Carnegie Mellon

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings

$ Vikings came in longboats from Scandinavia in 1000 AD

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came

$ Vikings came in longboats from Scandinavia in 1000 AD

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came in

$ Vikings came in longboats from Scandinavia in 1000 AD

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came in longboats

$ Vikings came in longboats from Scandinavia in 1000 AD

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came in longboats

$ Vikings came in longboats from Scandinavia in 1000 AD

Cycles, multiple roots, and non-projectivity
are all **permitted** by this model

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came in longboats

Only one parameter per dependency arc:

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came in longboats

Only one parameter per dependency arc:

$$p(\text{Vikings} \mid \text{came})$$

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came in longboats

Only one parameter per dependency arc:

$$p(\text{Vikings} \mid \text{came})$$

We cannot look at other dependency arcs, but we can condition on (properties of) the sentence:

$$p(\text{Vikings} \mid \text{came}, \mathit{direction} = \mathit{left}, \mathit{distance} = 1, \ldots)$$

$ Vikings came in longboats from Scandinavia in 1000 AD

Vikings came in longboats

We condition on direction:

$$p(\text{Vikings} \mid \text{came}, \mathit{direction} = \mathit{left})$$

("Concave Model A")

$ NNPS VBD IN NNS IN NNP IN CD NN

NNPS VBD IN NNS

Note: we've been using words in our examples, but in our model we follow standard practice and use gold POS tags

We condition on direction:

$$p(\mathrm{NNPS} \mid \mathrm{VBD}, \mathit{direction} = \mathit{left})$$

("Concave Model A")

$ NNPS VBD IN NNS IN NNP IN CD NN

NNPS VBD IN NNS

Model	Initializer	Accuracy*
Attach Right	N/A	31.7
DMV	Uniform	17.6
DMV	K&M	32.9
Concave Model A	Uniform	25.6

*Penn Treebank test set, sentences of all lengths

WSJ10 used for training

We condition on direction:

$$p(\text{NNPS} \mid \text{VBD}, \mathit{direction} = \mathit{left})$$

("Concave Model A")

Note:

IBM Model 1 is not *strictly* concave (Toutanova & Galley, 2011)

Model	Initializer	Accuracy*
Attach Right	N/A	31.7
DMV	Uniform	17.6
DMV	K&M	32.9
Concave Model A	Uniform	25.6

*Penn Treebank test set, sentences of all lengths

WSJ10 used for training

We condition on direction:

$$p(\mathrm{NNPS} \mid \mathrm{VBD}, \mathit{direction} = \mathit{left})$$

("Concave Model A")

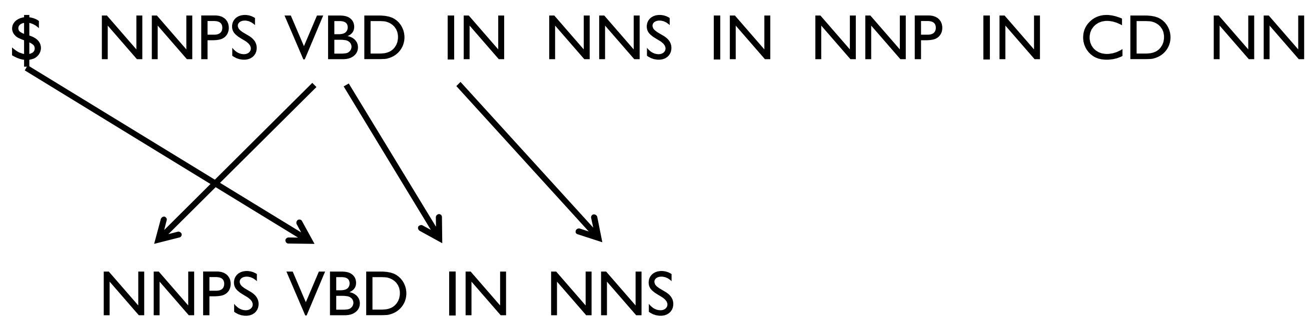
$ NNPS VBD IN NNS IN NNP IN CD NN

NNPS VBD IN NNS

We can also use hard constraints while preserving concavity:

The only tags that can align to $ are verbs
(Marecček & Žabokrtský, 2011; Naseem et al., 2010)

("Concave Model B")



Model	Initializer	Accuracy*
Attach Right	N/A	31.7
DMV	Uniform	17.6
DMV	K&M	32.9
Concave Model A	Uniform	25.6
Concave Model B	Uniform	28.6

*Penn Treebank test set, sentences of all lengths

WSJ10 used for training

Unsupervised learning in NLP $\xrightarrow{\text{(typically)}}$ non-convex optimization

Except IBM Model 1 for word alignment

(which has a concave log-likelihood function)

What models can we build without sacrificing concavity?

Can these concave models be useful?

As IBM Model 1 is used to initialize other word alignment models, we can use our concave models to initialize the DMV

As IBM Model 1 is used to initialize other word alignment models, we can use our concave models to initialize the DMV

Model	Initializer	Accuracy
Attach Right	N/A	31.7
DMV	Uniform	17.6
DMV	K&M	32.9
DMV	Concave Model A	34.4
DMV	Concave Model B	**43.0**

*Penn Treebank test set, sentences of all lengths

WSJ10 used for training

Carnegie Mellon

As IBM Model 1 is used to initialize other word alignment models, we can use our concave models to initialize the DMV

Model	Initializer	Accuracy*
DMV, trained on sentences of length ≤ 20	Concave Model B	53.1
Shared Logistic Normal (Cohen & Smith, 2009)	K&M	41.4
Posterior Regularization (Gillenwater et al., 2010)	K&M	53.3
LexTSG-DMV (Blunsom & Cohn, 2010)	K&M	55.7
Punctuation/UnsupTags (Spitkovsky et al., 2011), trained on sentences of length ≤ 45	K&M’	59.1

*Penn Treebank test set, sentences of all lengths

Multilingual Results
(averages across 18 languages)

Model	Initializer	Avg. Accuracy*	Avg. Log-Likelihood †
DMV	Uniform	25.7	-15.05
DMV	K&M	29.4	-14.84
DMV	Concave Model A	30.9	-14.93
DMV	Concave Model B	**35.5**	**-14.45**

* Sentences of all lengths from each test set

† Micro-averaged across sentences in all training sets
(used sentences ≤ 10 words for training)

Unsupervised learning in NLP $\xrightarrow{\text{(typically)}}$ non-convex optimization

Except IBM Model 1 for word alignment

(which has a concave log-likelihood function)

What models can we build without sacrificing concavity?

Can these concave models be useful?

Like word alignment, we can use simple, concave models to initialize more complex models for grammar induction

Thanks!