Fast Consensus Decoding over Translation Forests

Information Sciences Institute
USC School of Engineering

University of California
CAL NLP
Berkeley

John DeNero, David Chiang, and Kevin Knight
denero@berkeley.edu, {chiang, knight}@isi.edu

Overview

- Minimum Bayes risk (MBR) decoding tends to improve over model-best (Viterbi) decoding
- With a *linear* loss function, MBR is efficient
- Our variant, *fast consensus decoding*, is efficient even with non-linear loss functions (e.g., BLEU)

The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z}\exp\Big(\ \vec{\lambda} \cdot \vec{\varphi}(e, f)\ \Big)$$

The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z} \exp\Big(\vec{\lambda} \cdot \vec{\varphi}(e, f) \Big)$$

Minimizing Bayes risk

$$R(e) = 1 - \mathbb{E}_{P(e'|f)}[S(e; e')]$$

The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z}\exp\Big(\ \vec{\lambda}\cdot\vec{\varphi}(e,f)\ \Big)$$

Minimizing Bayes risk

$$R(e) = 1 - \mathbb{E}_{P(e'|f)}[S(e;e')]$$

Sentence similarity evaluation measure (e.g., BLEU)

The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z} \exp\Big(\vec{\lambda} \cdot \vec{\varphi}(e, f) \Big)$$

Minimizing Bayes risk

$$R(e) = 1 - \mathbb{E}_{P(e'|f)}[S(e;e')]$$

Our model's posterior distribution

Sentence similarity evaluation measure (e.g., BLEU)

The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z} \exp\Big(\vec{\lambda} \cdot \vec{\varphi}(e, f) \Big)$$

Minimizing Bayes risk

$$R(e) = \underbrace{1 -}_{\text{Flip sign to get “loss”}} \underbrace{\mathbb{E}_{P(e'|f)}}_{\text{Our model’s posterior distribution}} \underbrace{[S(e; e')]}_{\text{Sentence similarity evaluation measure (e.g., BLEU)}}$$

The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z}\exp\Big(\ \vec{\lambda}\cdot\vec{\varphi}(e,f)\ \Big)$$

Minimizing Bayes risk == Maximizing expected similarity

$$R(e) = 1 - \mathbb{E}_{P(e'|f)}[S(e;e')]$$

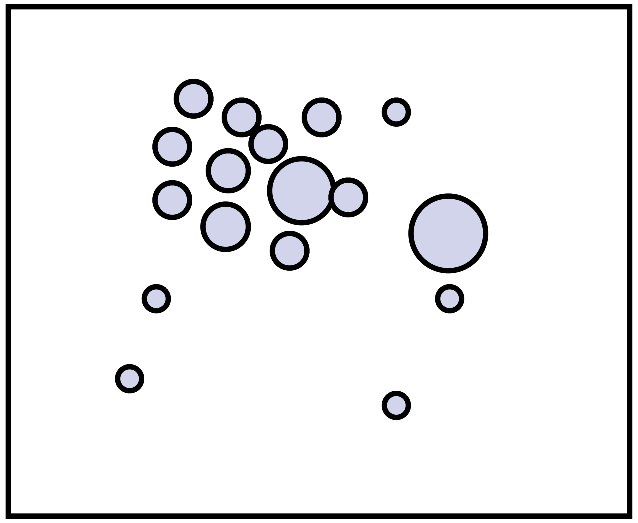
The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z} \exp\Big(\vec{\lambda} \cdot \vec{\varphi}(e, f) \Big)$$

Minimizing Bayes risk == Maximizing expected similarity

$$R(e) = 1 - \mathbb{E}_{P(e'|f)}[S(e; e')]$$



Points:
Translations

Size:
Posterior

Distance:
Similarity

Dramatization

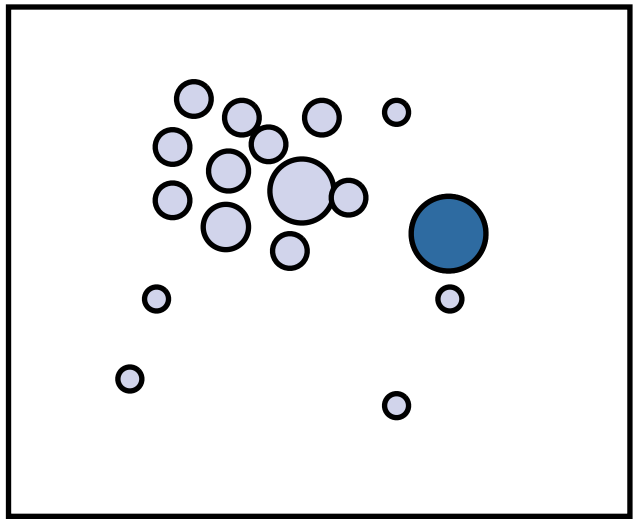
The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z}\exp\left(\boxed{\vec{\lambda}\cdot\vec{\varphi}(e,f)}\right)$$

Minimizing Bayes risk == Maximizing expected similarity

$$R(e) = 1 - \mathbb{E}_{P(e'|f)}[S(e;e')]$$



Points:
Translations

Size:
Posterior

Distance:
Similarity

Dramatization

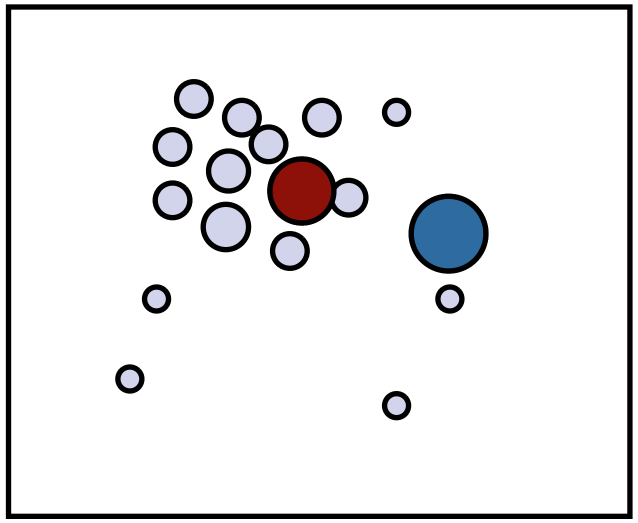
The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

$$P(e|f) = \frac{1}{Z} \exp\left(\boxed{\vec{\lambda} \cdot \vec{\varphi}(e, f)} \right)$$

Minimizing Bayes risk == Maximizing expected similarity

$$R(e) = 1 - \boxed{\mathbb{E}_{P(e'|f)}[S(e; e')]}$$



Points:
Translations

Size:
Posterior

Distance:
Similarity

Dramatization

The Minimum Bayes Risk Objective

MT models induce posterior distributions over outputs

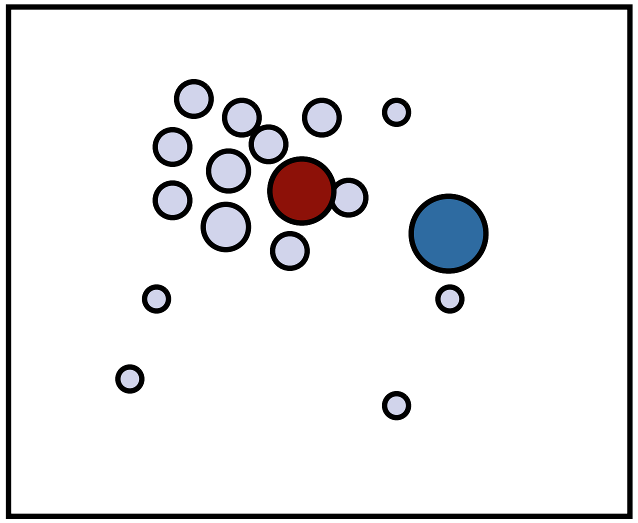
$$P(e|f) = \frac{1}{Z}\exp\left(\vec{\lambda}\cdot\vec{\varphi}(e,f)\right)$$

Minimizing Bayes risk == Maximizing expected similarity

$$R(e) = 1 - \mathbb{E}_{P(e'|f)}[S(e;e')]$$

E.g., Unigram Precision:

$$\frac{\text{Number of word types in e and e'}}{\text{Number of word types in e}}$$



Points: Translations

Size: Posterior

Distance: Similarity

Dramatization

Maximizing Expected Similarity over K-Best Lists

Maximizing Expected Similarity over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

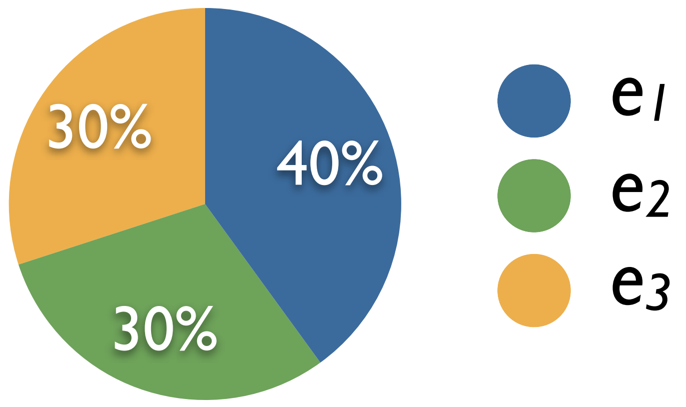
Maximizing Expected Similarity over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

$$\frac{\exp\left(\vec{\lambda} \cdot \vec{\varphi}(e, f)\right)}{Z}$$

Exponentiate & Normalize



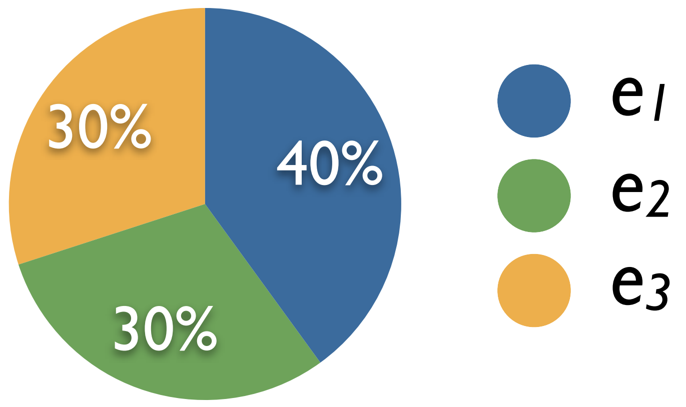
Maximizing Expected Similarity over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

$$\frac{\exp\left(\vec{\lambda}\cdot\vec{\varphi}(e,f)\right)}{Z}$$

Exponentiate & Normalize



Compute K² Pairwise Similarities

References

	e_1	e_2	e_3
e_1	3/3	1/2	0/2
e_2	1/3	2/2	1/2
e_3	0/3	1/2	2/2

Hypotheses

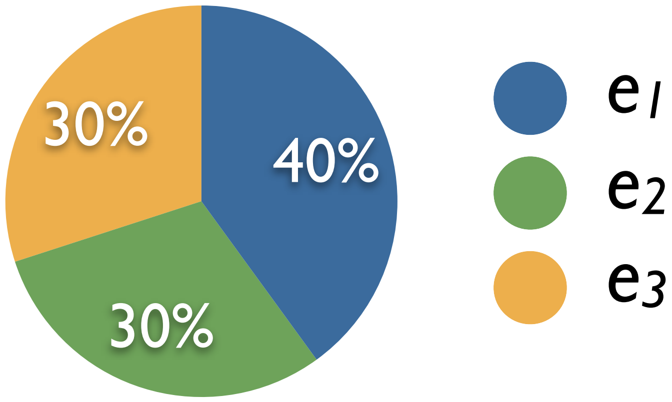
Maximizing Expected Similarity over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

$$\frac{\exp\left(\vec{\lambda}\cdot\vec{\varphi}(e,f)\right)}{Z}$$

Exponentiate & Normalize

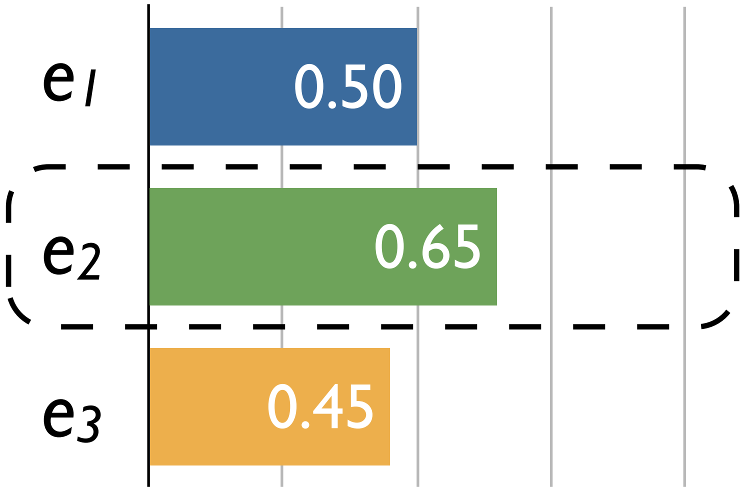


Compute K^2 Pairwise Similarities

	Hypotheses		
References	e_1	e_2	e_3
e_1	3/3	1/2	0/2
e_2	1/3	2/2	1/2
e_3	0/3	1/2	2/2

$$\sum_{e'} S(e;e')\cdot P(e'|f)$$

Max over Expectations



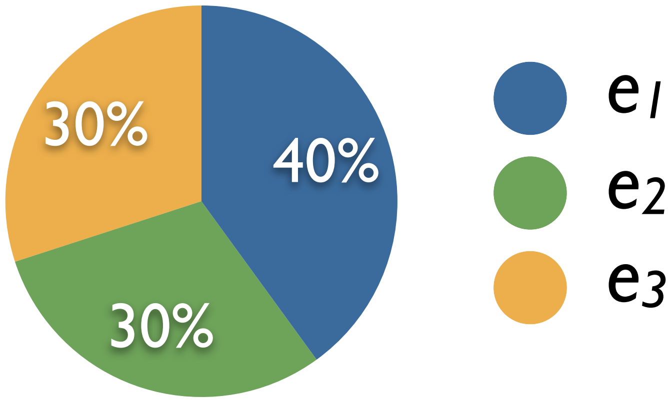
Maximizing Expected Similarity over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

$$\frac{\exp\left(\vec{\lambda}\cdot\vec{\varphi}(e,f)\right)}{Z}$$

Exponentiate & Normalize



Compute K^2 Pairwise Similarities

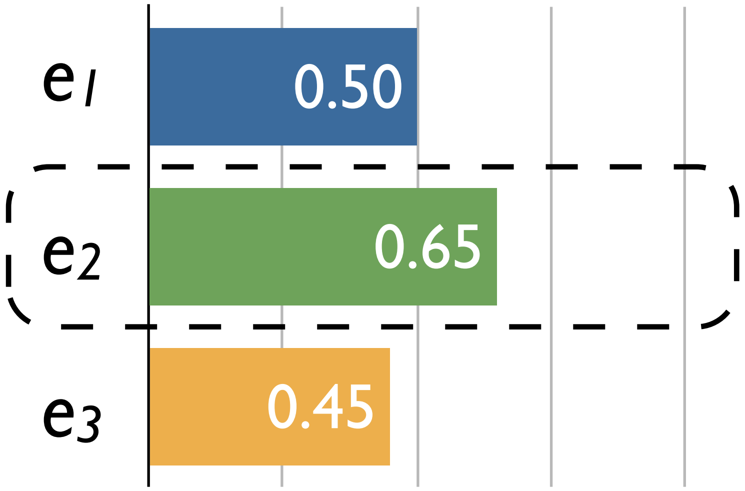
References

	e_1	e_2	e_3
e_1	3/3	1/2	0/2
e_2	1/3	2/2	1/2
e_3	0/3	1/2	2/2

Hypotheses

$$\sum_{e'} S(e;e')\cdot P(e'|f)$$

Max over Expectations



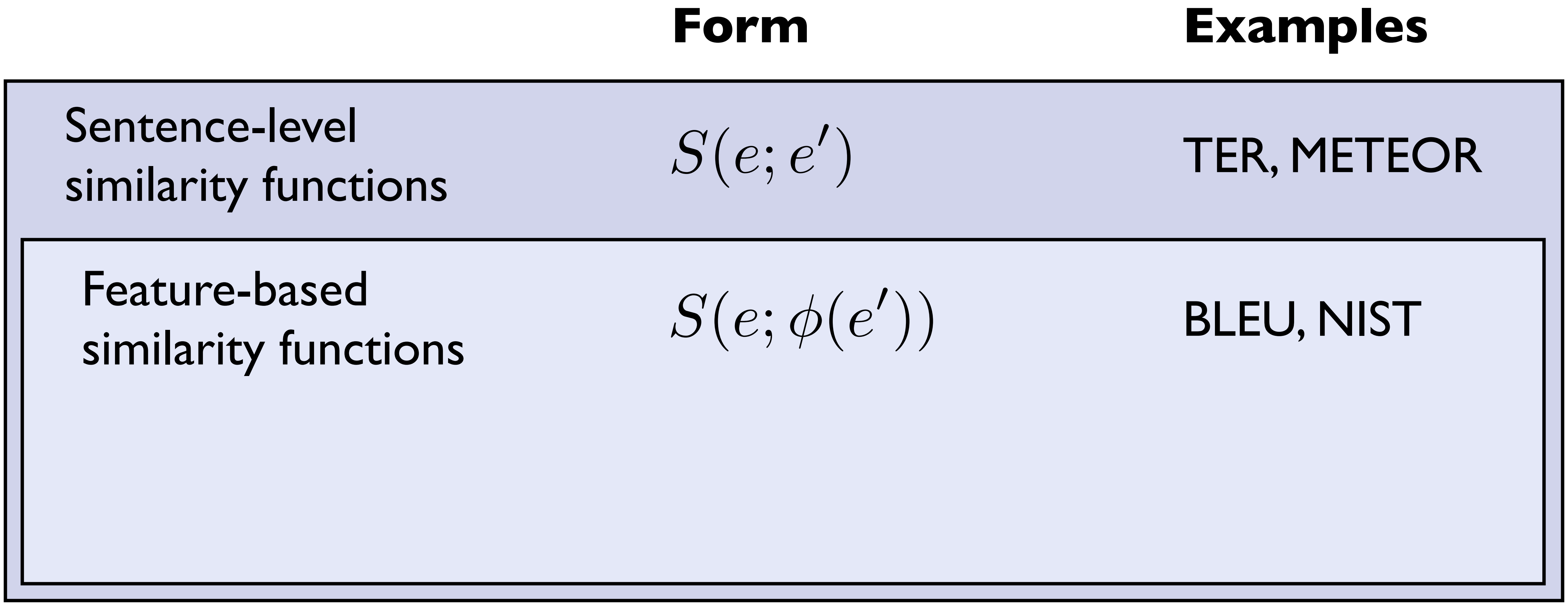
Restricted Families of Similarity Functions

	Form	Examples
Sentence-level similarity functions	$S(e; e')$	TER, METEOR

Restricted Families of Similarity Functions

	Form	**Examples**
Sentence-level similarity functions	$S(e; e')$	TER, METEOR
Feature-based similarity functions	$S(e; \phi(e'))$	BLEU, NIST

Restricted Families of Similarity Functions



	Form	Examples
Sentence-level similarity functions	$S(e; e')$	TER, METEOR
Feature-based similarity functions	$S(e; \phi(e'))$	BLEU, NIST

BLEU:

$$\exp\left[\left(1 - \frac{|e'|}{|e|}\right)_{-} + \frac{1}{4}\sum_{n=1}^{4} \ln \frac{\sum_{t \in T_n} \min(c(e,t), c(e',t))}{\sum_{t \in T_n} c(e,t)}\right]$$

Restricted Families of Similarity Functions

	Form	**Examples**
Sentence-level similarity functions	$S(e; e')$	TER, METEOR
Feature-based similarity functions	$S(e; \phi(e'))$	BLEU, NIST

BLEU:

Features $\phi(e')$ are counts of *n*-grams in e'

$$\exp\left[\left(1 - \frac{|e'|}{|e|}\right)_{-} + \frac{1}{4}\sum_{n=1}^{4} \ln \frac{\sum_{t \in T_n} \min(c(e,t), \overline{c(e',t)})}{\sum_{t \in T_n} c(e,t)}\right]$$

Restricted Families of Similarity Functions

	Form	**Examples**
Sentence-level similarity functions	$S(e; e')$	TER, METEOR
Feature-based similarity functions	$S(e; \phi(e'))$	BLEU, NIST
Linear functions of features	$\omega(e) \cdot \phi(e')$	Unigram Precision

Restricted Families of Similarity Functions

	Form	**Examples**
Sentence-level similarity functions	$S(e; e')$	TER, METEOR
Feature-based similarity functions	$S(e; \phi(e'))$	BLEU, NIST
Linear functions of features	$\omega(e) \cdot \phi(e')$	Unigram Precision

Unigram Precision:

$$\sum_{t \in T_1} \left(\frac{\delta(t, e)}{\underbrace{|\{e\}|}_{\text{unique word types in } e}} \right) \cdot \delta(t, e')$$

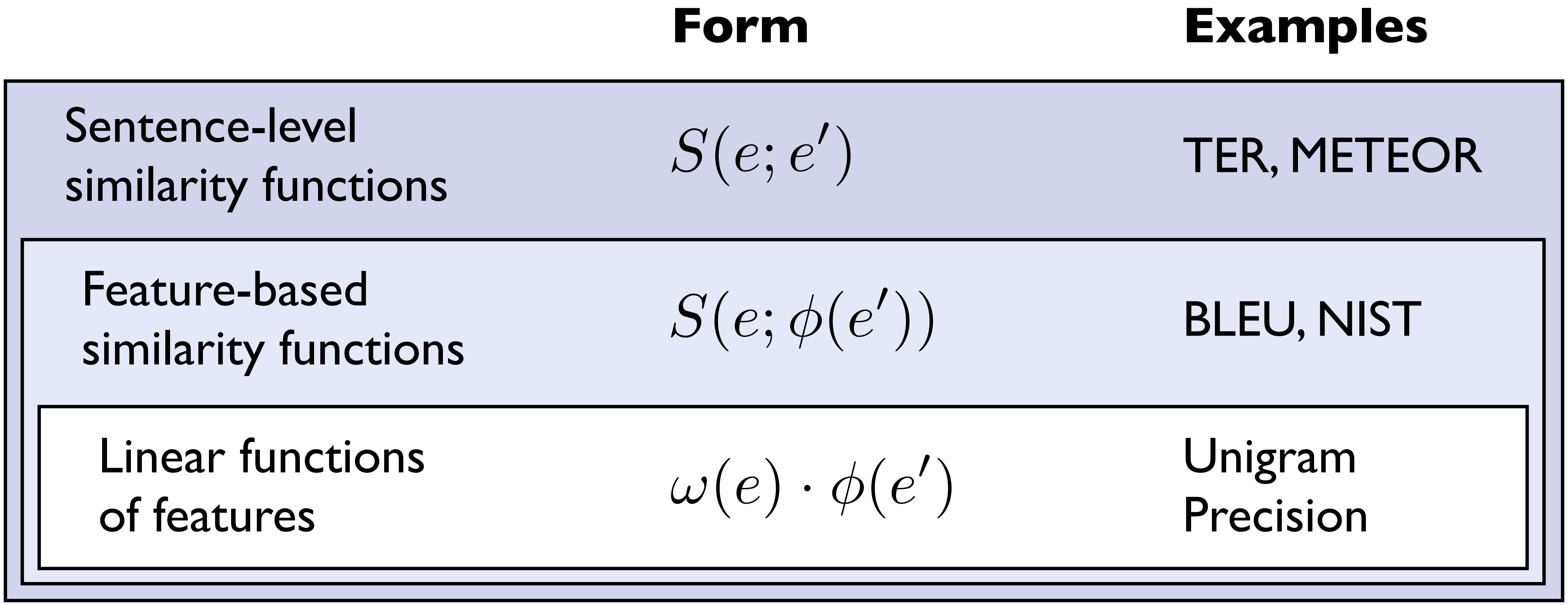
Restricted Families of Similarity Functions

	Form	**Examples**
Sentence-level similarity functions	$S(e; e')$	TER, METEOR
Feature-based similarity functions	$S(e; \phi(e'))$	BLEU, NIST
Linear functions of features	$\omega(e) \cdot \phi(e')$	Unigram Precision

Unigram Precision:

$$\sum_{t\in T_1}\left(\frac{\delta(t,e)}{\underbrace{|\{e\}|}_{\text{unique word types in } e}}\right)\cdot\overbrace{\delta(t,e')}^{\phi_t(e')}$$

Restricted Families of Similarity Functions



	Form	Examples
Sentence-level similarity functions	$S(e; e')$	TER, METEOR
Feature-based similarity functions	$S(e; \phi(e'))$	BLEU, NIST
Linear functions of features	$\omega(e) \cdot \phi(e')$	Unigram Precision

Unigram Precision:

$$\sum_{t\in T_1} \overbrace{\left(\frac{\delta(t,e)}{\underbrace{|\{e\}|}_{\text{unique word types in e}}}\right)}^{\omega_t(e)} \cdot \overbrace{\delta(t,e')}^{\phi_t(e')}$$

Fast Expected Similarity for Linear Functions

Linear functions of local features: $S(e; e') = \omega(e) \cdot \phi(e')$

Fast Expected Similarity for Linear Functions

Linear functions of local features: $S(e;e') = \omega(e) \cdot \phi(e')$

Linearity of expectations: $\mathbb{E}_{P(X)}\left[c \cdot X\right] = c \cdot \mathbb{E}_{P(X)}\left[X\right]$

Fast Expected Similarity for Linear Functions

Linear functions of local features: $S(e;e') = \omega(e) \cdot \phi(e')$

Linearity of expectations: $\mathbb{E}_{P(X)}\left[c \cdot X\right] = c \cdot \mathbb{E}_{P(X)}\left[X\right]$

$$\mathbb{E}_{e'|f}[S(e;e')] \;=\; \mathbb{E}_{e'|f}[\omega(e) \cdot \phi(e')] \;=\; \omega(e) \cdot \mathbb{E}\left[\phi(e')\right]$$

Fast Expected Similarity for Linear Functions

Linear functions of local features: $S(e;e') = \omega(e) \cdot \phi(e')$

Linearity of expectations: $\mathbb{E}_{P(X)}\left[c \cdot X\right] = c \cdot \mathbb{E}_{P(X)}\left[X\right]$

$$\mathbb{E}_{e'|f}[S(e;e')] \ = \ \mathbb{E}_{e'|f}[\omega(e) \cdot \phi(e')] \ = \ \omega(e) \cdot \mathbb{E}\left[\phi(e')\right]$$

Maximizing expected similarity equals: $\arg\max_{e} S(e; \mathbb{E}\left[\phi(e')\right])$

Fast Expected Similarity for Linear Functions

Linear functions of local features: $S(e; e') = \omega(e) \cdot \phi(e')$

Linearity of expectations: $\mathbb{E}_{P(X)}\left[c \cdot X\right] = c \cdot \mathbb{E}_{P(X)}\left[X\right]$

$$\mathbb{E}_{e'|f}[S(e; e')] \;=\; \mathbb{E}_{e'|f}[\omega(e) \cdot \phi(e')] \;=\; \omega(e) \cdot \mathbb{E}\left[\phi(e')\right]$$

Maximizing expected similarity equals: $\arg\max_{e} S(e; \underbrace{\mathbb{E}\left[\phi(e')\right]}_{\text{Average sentence in feature space}})$

Fast Expected Similarity for Linear Functions

Linear functions of local features: $S(e; e') = \omega(e) \cdot \phi(e')$

Linearity of expectations: $\mathbb{E}_{P(X)}[c \cdot X] = c \cdot \mathbb{E}_{P(X)}[X]$

$$\mathbb{E}_{e'|f}[S(e; e')] \;=\; \mathbb{E}_{e'|f}[\omega(e) \cdot \phi(e')] \;=\; \omega(e) \cdot \mathbb{E}[\phi(e')]$$

Maximizing expected similarity equals:

$$\arg\max_e \underbrace{S(e; \underbrace{\mathbb{E}[\phi(e')]}_{\text{Average sentence in feature space}})}_{\text{Similarity to the average sentence}}$$

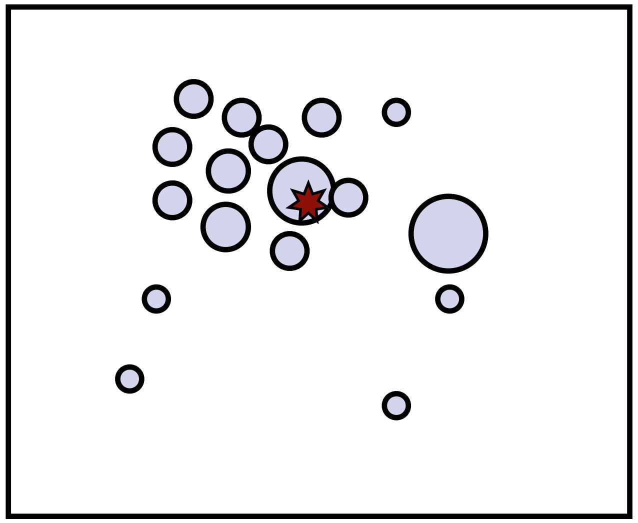
Fast Expected Similarity for Linear Functions

Linear functions of local features: $S(e; e') = \omega(e) \cdot \phi(e')$

Linearity of expectations: $\mathbb{E}_{P(X)}\left[c \cdot X\right] = c \cdot \mathbb{E}_{P(X)}\left[X\right]$

$$\mathbb{E}_{e'|f}[S(e;e')] \;=\; \mathbb{E}_{e'|f}[\omega(e) \cdot \phi(e')] \;=\; \omega(e) \cdot \mathbb{E}\left[\phi(e')\right]$$

Maximizing expected similarity equals: $\arg\max_{e} \underbrace{S(e; \underbrace{\mathbb{E}\left[\phi(e')\right]}_{\text{Average sentence in feature space}})}_{\text{Similarity to the average sentence}}$



Points:
$\phi(e)$

Size:
$P(e|f)$

Star:
$\mathbb{E}\left[\phi(e)\right]$

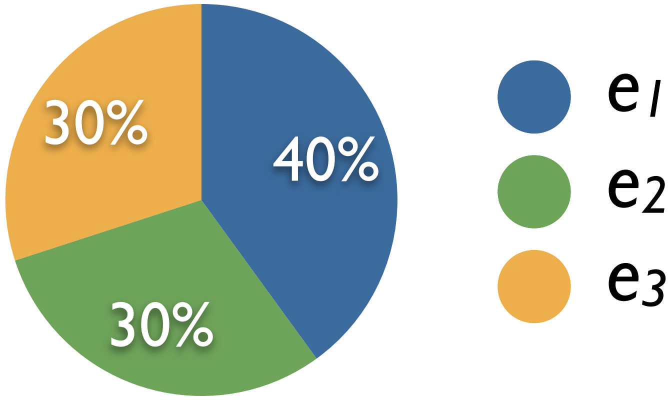
Fast Consensus Decoding over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

$$\frac{\exp\left(\vec{\lambda}\cdot\vec{\varphi}(e,f)\right)}{Z}$$

Exponentiate & Normalize



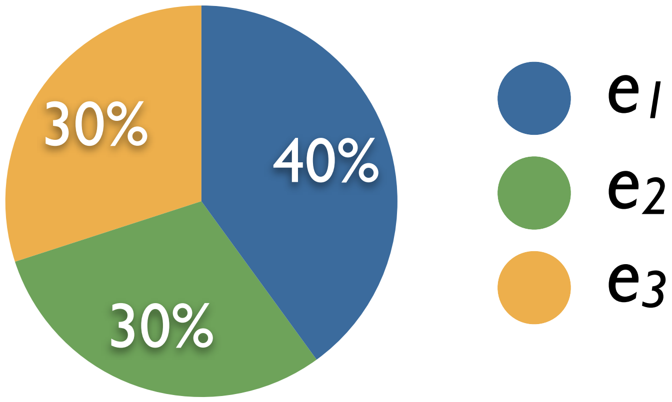
Fast Consensus Decoding over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

$$\frac{\exp\left(\vec{\lambda}\cdot\vec{\varphi}(e,f)\right)}{Z}$$

Exponentiate & Normalize



Computed Feature Expectations

	e_1	e_2	e_3	
forest	0	1	1	0.6
forests	1	0	0	0.4
decoding	1	1	0	0.7
decoders	0	0	1	0.3
of	1	0	0	0.4

$$\mathbb{E}\left[\phi(e')\right]$$

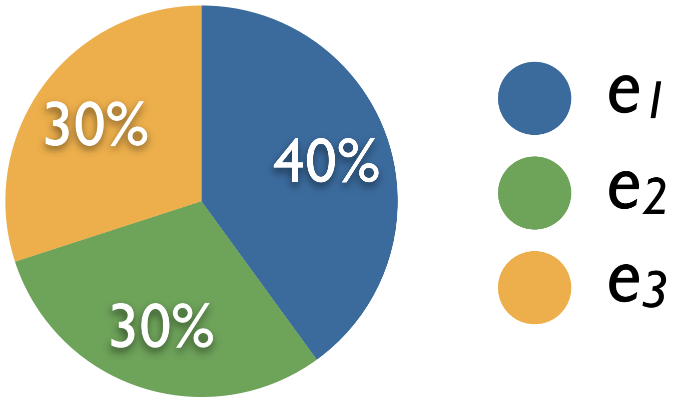
Fast Consensus Decoding over K-Best Lists

Decode to Create a K-Best List

e_1 :	*decoding of forests*	-0.22
e_2 :	*forest decoding*	-0.51
e_3 :	*forest decoders*	-0.51

$$\frac{\exp\left(\vec{\lambda}\cdot\vec{\varphi}(e,f)\right)}{Z}$$

Exponentiate & Normalize



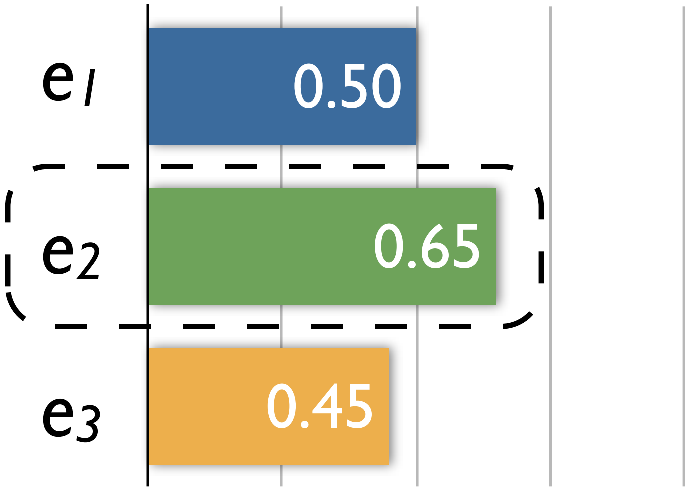
Computed Feature Expectations

	e_1	e_2	e_3	
forest	0	1	1	0.6
forests	1	0	0	0.4
decoding	1	1	0	0.7
decoders	0	0	1	0.3
of	1	0	0	0.4

$$\mathbb{E}\left[\phi(e')\right]$$

$$S(e;\mathbb{E}\left[\phi(e')\right])$$

Max over Similarities



Fast Consensus Decoding with BLEU

$$\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

Fast Consensus Decoding with BLEU

$$\arg\max_{e} \mathrm{BLEU}(e; \underline{\mathbb{E}[\phi(e')]})$$

Expected n-gram counts

Fast Consensus Decoding with BLEU

$$\arg\max_{e} \mathrm{BLEU}(e; \underline{\mathbb{E}[\phi(e')]})$$

Expected n-gram counts

$$= \quad \arg\max_{e}$$

$$\exp\left[\left(1 - \frac{\mathbb{E}[\,|e'|\,]}{|e|}]\right)_{-} + \frac{1}{4}\sum_{n=1}^{4} \ln \frac{\sum_{t \in T_n} \min(c(e,t), \mathbb{E}\,[c(e',t)])}{\sum_{t \in T_n} c(e,t)}\right]$$

Length penalty computed relative to the expected length of the output

N-gram counts are clipped by the expected count of each n-gram

Fast Consensus Decoding with BLEU

$$\arg\max_e \mathrm{BLEU}(e; \underline{\mathbb{E}[\phi(e')]})$$

Expected n-gram counts

$$= \quad \arg\max_e$$

$$\exp\left[\left(1 - \frac{\mathbb{E}[\ |e'|\ ]}{|e|}]\right)_- + \frac{1}{4}\sum_{n=1}^{4}\ln\frac{\sum_{t\in T_n}\min(c(e,t), \mathbb{E}\left[c(e',t)\right])}{\sum_{t\in T_n} c(e,t)}\right]$$

Length penalty computed relative to the expected length of the output

N-gram counts are clipped by the expected count of each n-gram

On 1000-best lists:

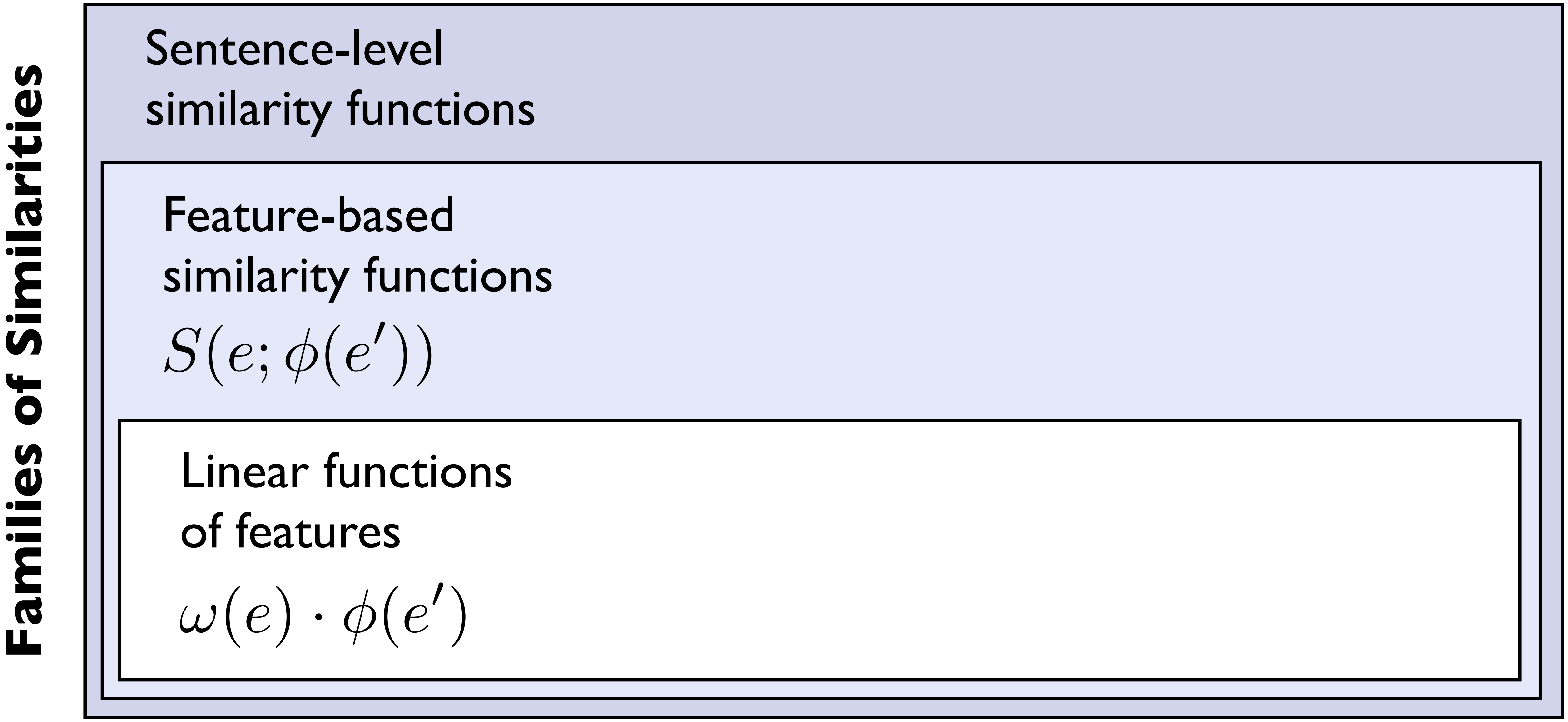
80 times faster than MBR with nearly identical improvements (within 0.1 BLEU in all test conditions)

Consensus Decoding Algorithm Landscape

	Expected Similarity (min. Bayes risk)	**vs**	Fast Consensus
$\arg\max_{e}$	$\mathbb{E}\left[S(e;e')\right]$		$S\left(e;\mathbb{E}\left[\phi(e')\right]\right)$

Consensus Decoding Algorithm Landscape

	Expected Similarity (min. Bayes risk)	**vs**	Fast Consensus
$\arg\max_{e}$	$\mathbb{E}\left[S(e;e')\right]$		$S\left(e;\mathbb{E}\left[\phi(e')\right]\right)$



Families of Similarities

Sentence-level
similarity functions

Feature-based
similarity functions

$S(e;\phi(e'))$

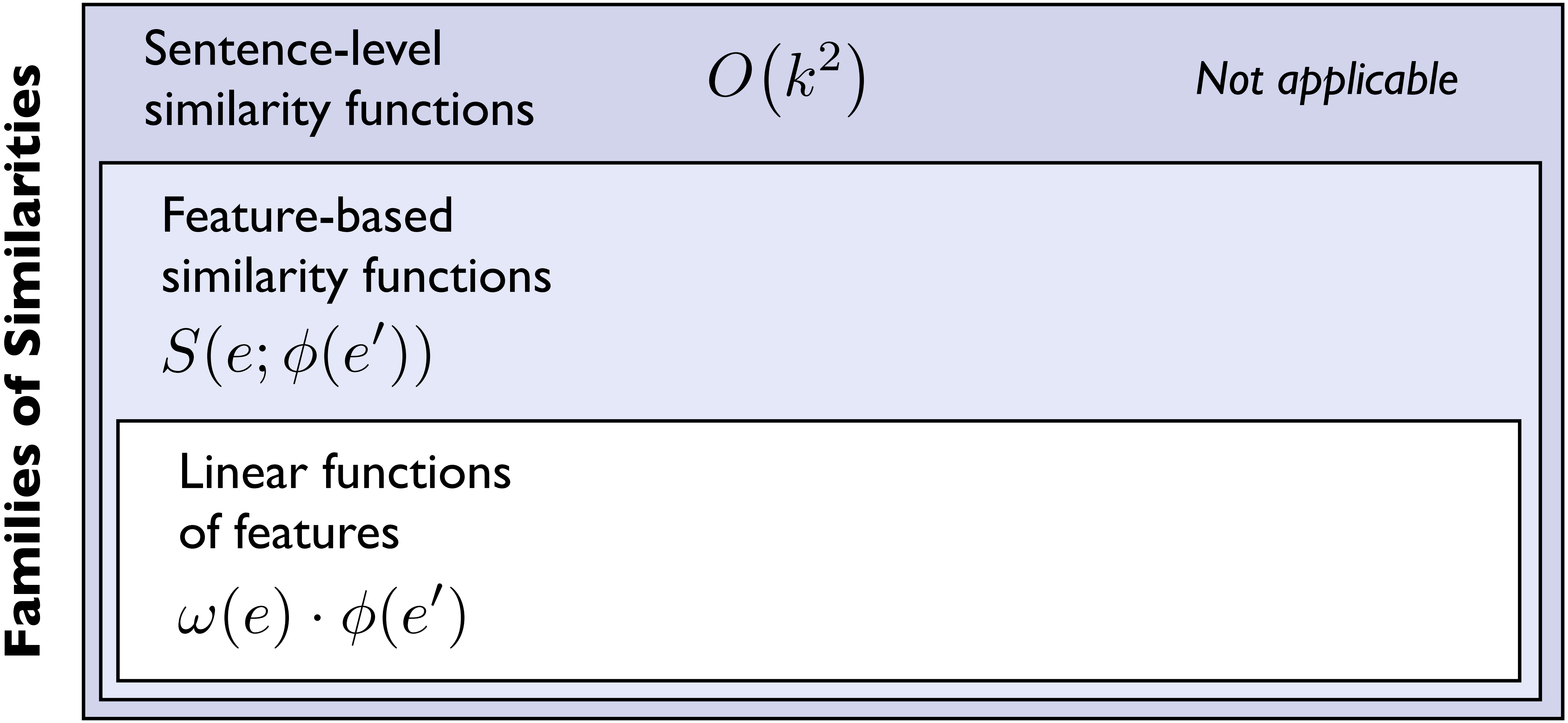
Linear functions
of features

$\omega(e)\cdot\phi(e')$

Consensus Decoding Algorithm Landscape

	Expected Similarity (min. Bayes risk)	**vs**	Fast Consensus
$\arg\max_{e}$	$\mathbb{E}\left[S(e;e')\right]$		$S\left(e;\mathbb{E}\left[\phi(e')\right]\right)$

Families of Similarities



Sentence-level similarity functions

$O\left(k^2\right)$

Not applicable

Feature-based similarity functions

$S(e;\phi(e'))$

Linear functions of features

$\omega(e)\cdot\phi(e')$

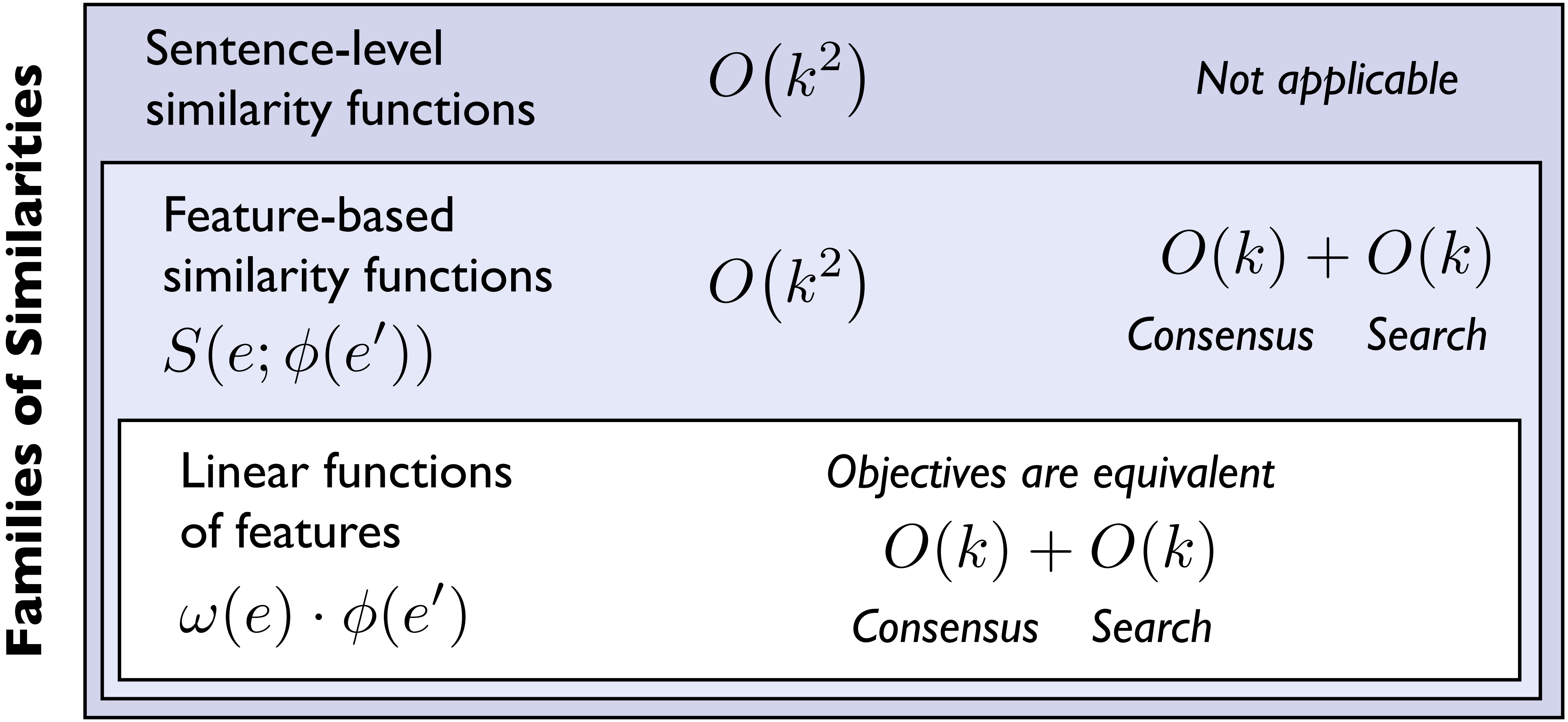
Consensus Decoding Algorithm Landscape

Families of Similarities

	Expected Similarity (min. Bayes risk)	**vs**	Fast Consensus
$\arg\max_{e}$	$\mathbb{E}\left[S(e;e')\right]$		$S\left(e;\mathbb{E}\left[\phi(e')\right]\right)$
Sentence-level similarity functions	$O\left(k^2\right)$		*Not applicable*
Feature-based similarity functions $S(e;\phi(e'))$	$O\left(k^2\right)$		$O(k)+O(k)$ *Consensus* *Search*
Linear functions of features $\omega(e)\cdot\phi(e')$			

Consensus Decoding Algorithm Landscape

	Expected Similarity (min. Bayes risk)	**vs**	Fast Consensus
$\arg\max_{e}$	$\mathbb{E}\left[S(e;e')\right]$		$S\left(e;\mathbb{E}\left[\phi(e')\right]\right)$



Families of Similarities

Sentence-level similarity functions — $O\left(k^2\right)$ — *Not applicable*

Feature-based similarity functions $S(e;\phi(e'))$ — $O\left(k^2\right)$ — $O(k) + O(k)$ *Consensus* *Search*

Linear functions of features $\omega(e)\cdot\phi(e')$ — *Objectives are equivalent* $O(k) + O(k)$ *Consensus* *Search*

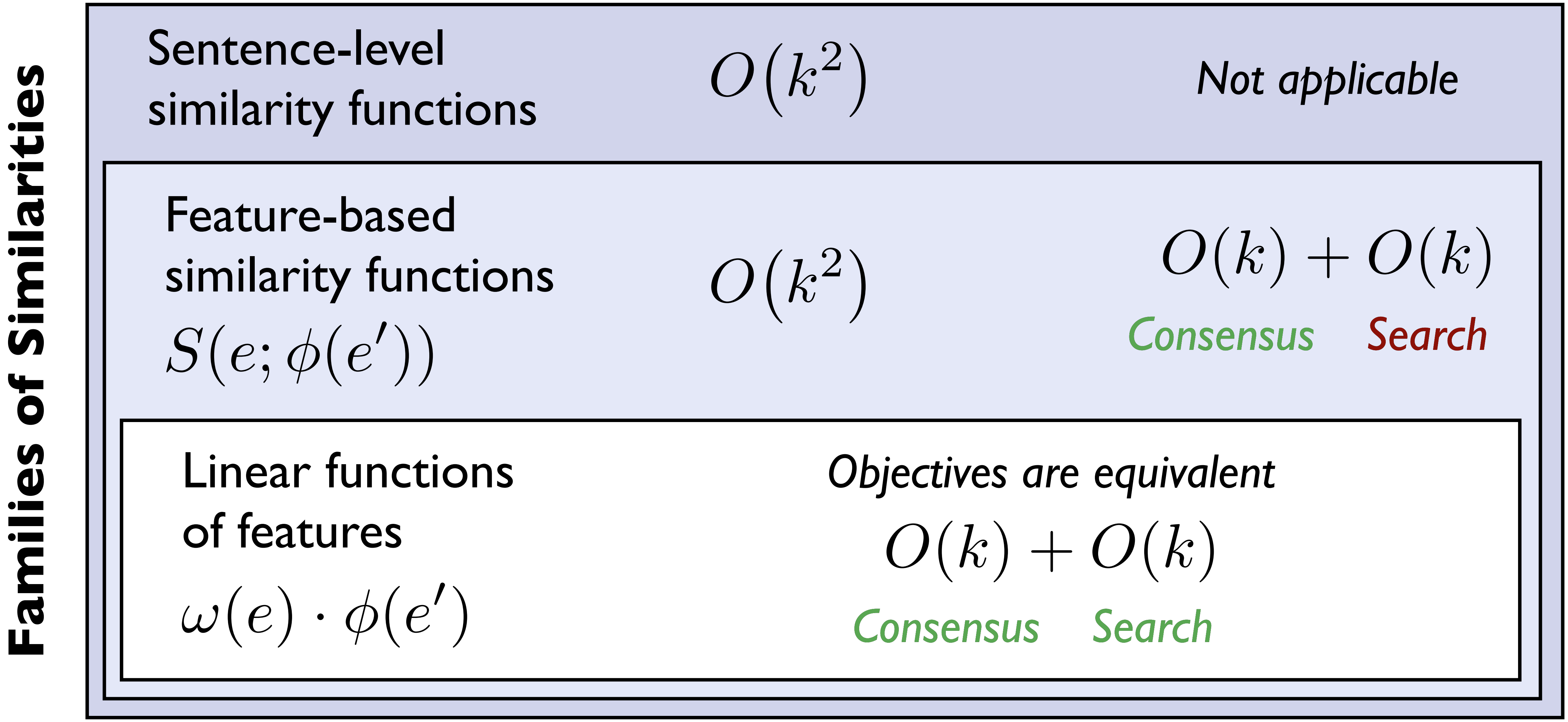
Consensus Decoding Algorithm Landscape

	Expected Similarity (min. Bayes risk)	**vs**	Fast Consensus
$\arg\max_{e}$	$\mathbb{E}\left[S(e; e')\right]$		$S\left(e; \mathbb{E}\left[\phi(e')\right]\right)$

Families of Similarities

Sentence-level similarity functions	$O\left(k^2\right)$	*Not applicable*
Feature-based similarity functions $S(e; \phi(e'))$	$O\left(k^2\right)$	$O(k) + O(k)$ *Consensus* *Search*
Linear functions of features $\omega(e) \cdot \phi(e')$	*Objectives are equivalent* $O(k) + O(k)$ *Consensus* *Search*	

Consensus Decoding Algorithm Landscape



	Expected Similarity (min. Bayes risk)	vs	Fast Consensus
$\arg\max_{e}$	$\mathbb{E}\left[S(e;e')\right]$		$S\left(e;\mathbb{E}\left[\phi(e')\right]\right)$
Sentence-level similarity functions	$O\left(k^2\right)$		*Not applicable*
Feature-based similarity functions $S(e;\phi(e'))$	$O\left(k^2\right)$		$O(k) + O(k)$ Consensus Search
Linear functions of features $\omega(e) \cdot \phi(e')$	*Objectives are equivalent* $O(k) + O(k)$ Consensus Search		

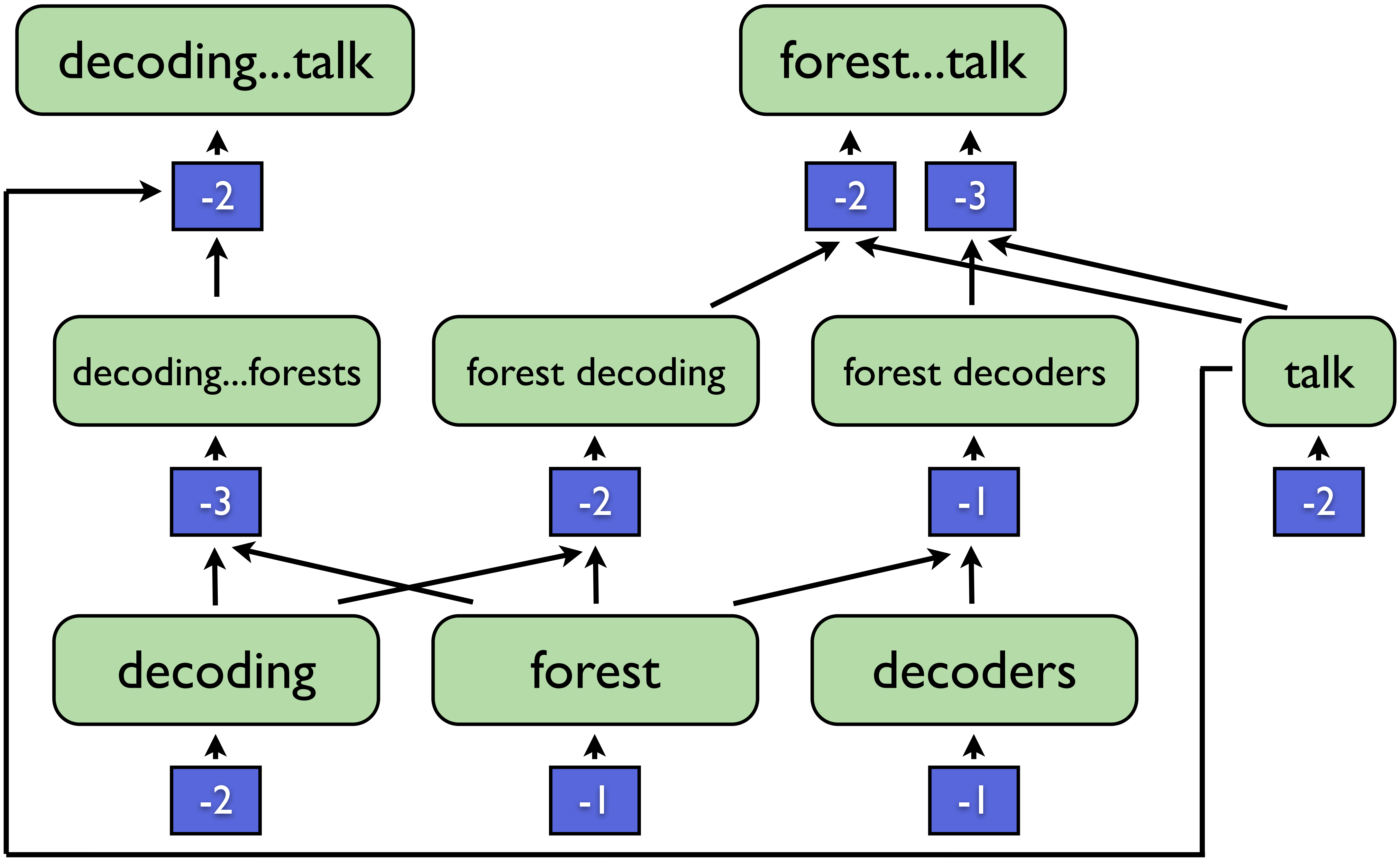
Families of Similarities

N-gram Expectations from Forests

decoding talk

forest talk

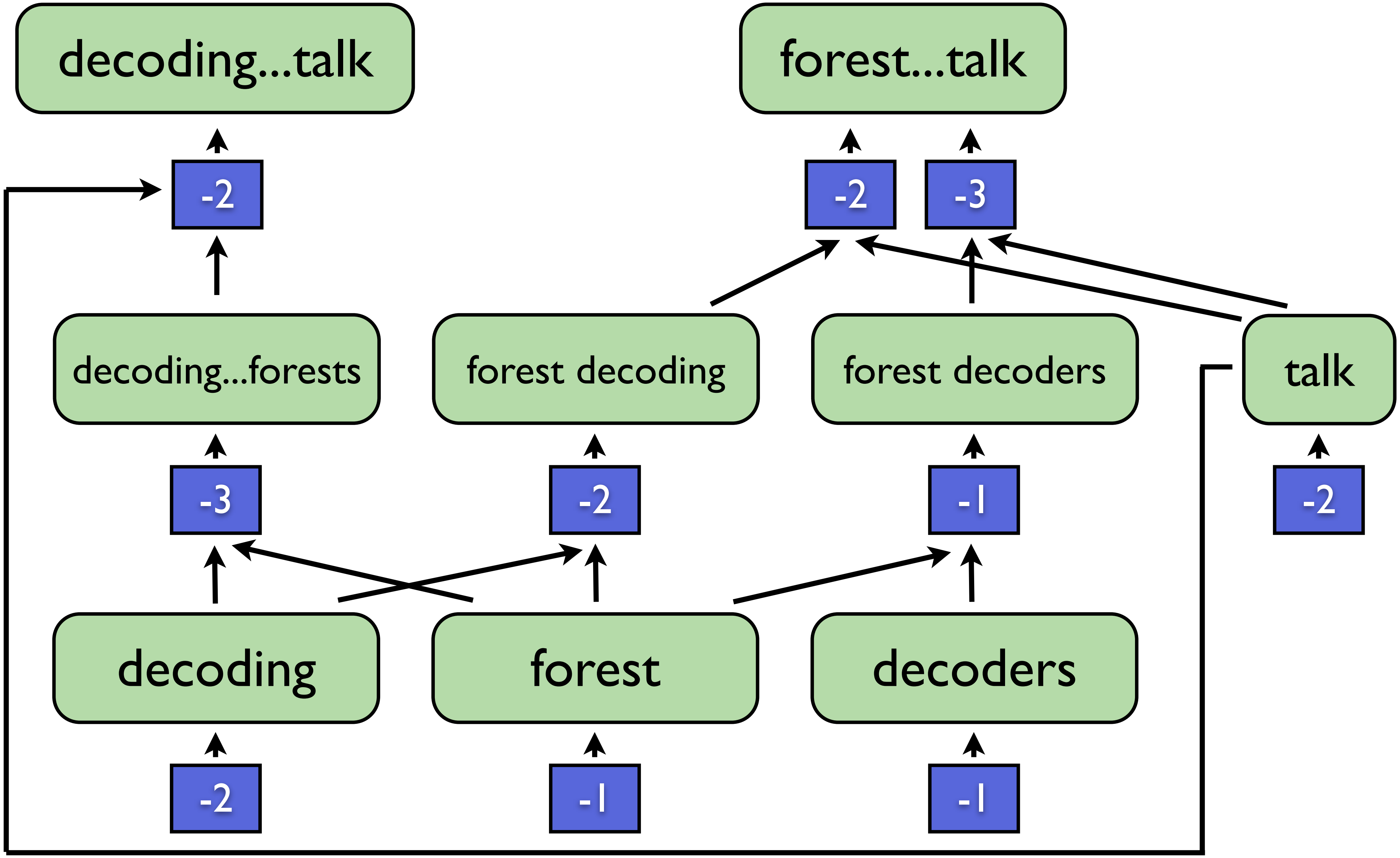
N-gram Expectations from Forests



N-gram Expectations from Forests

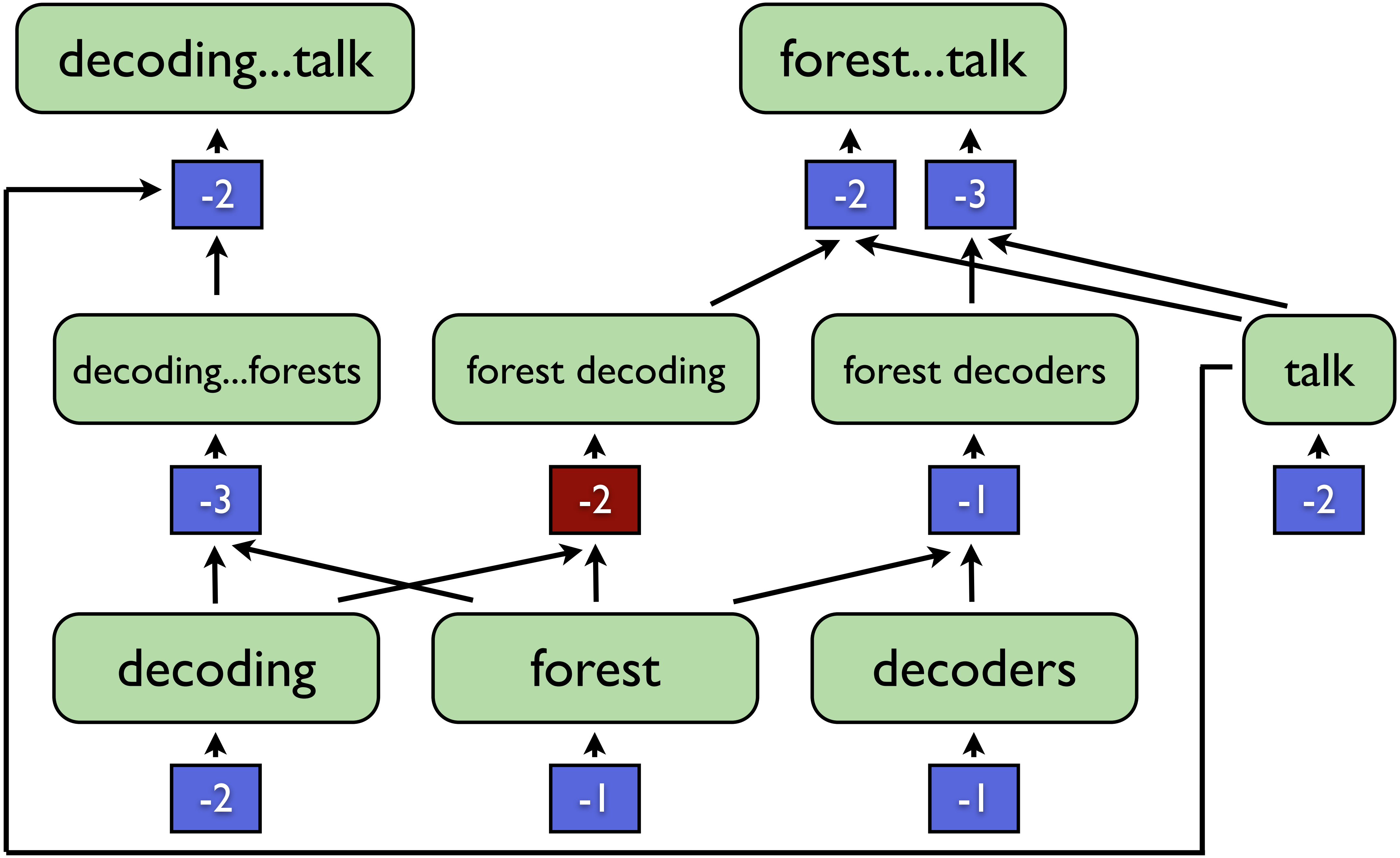
"decoding of forests talk"

"forest {decoding,decoders} talk"



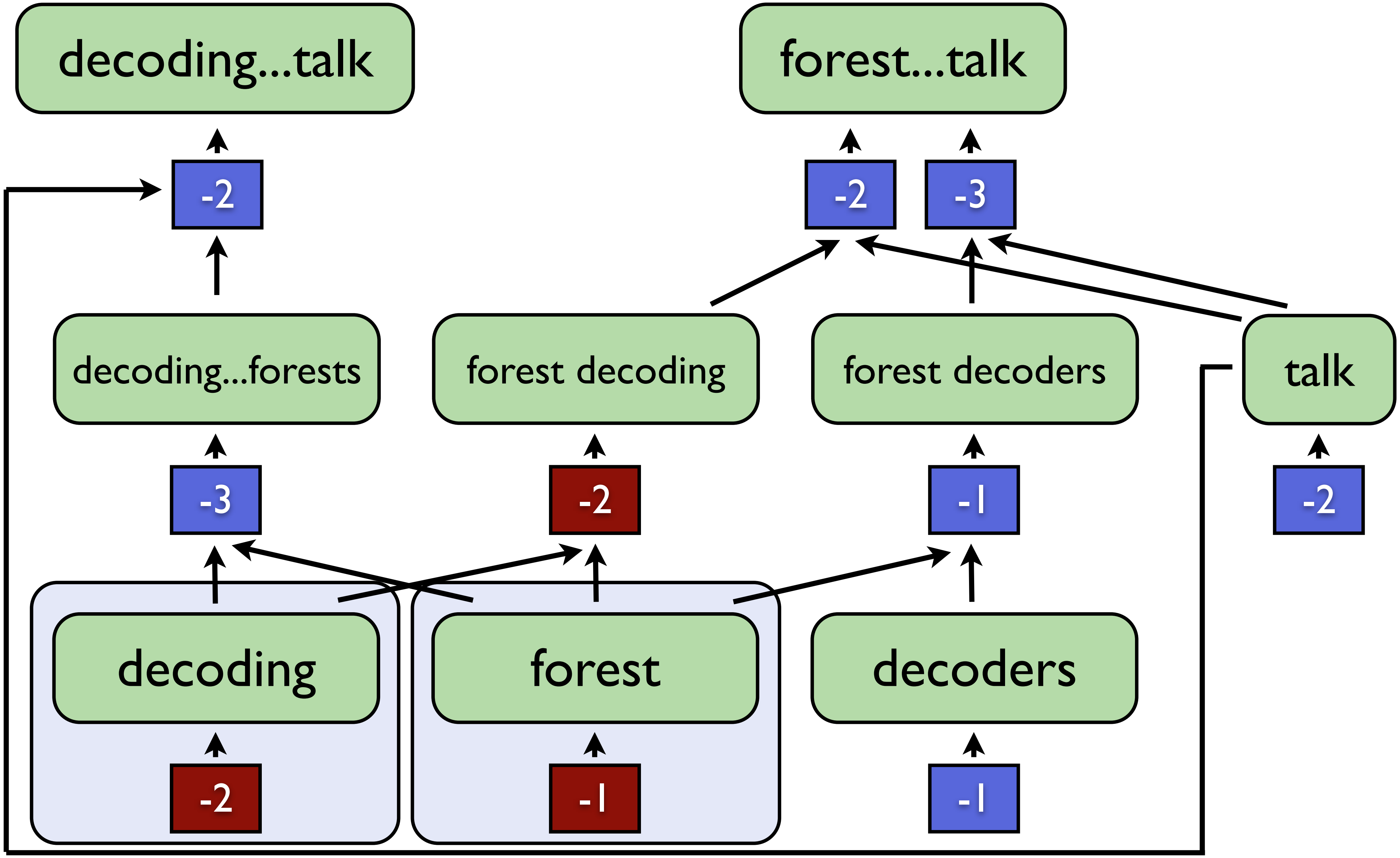
N-gram Expectations from Forests

Step 1: Compute posterior probability of each edge



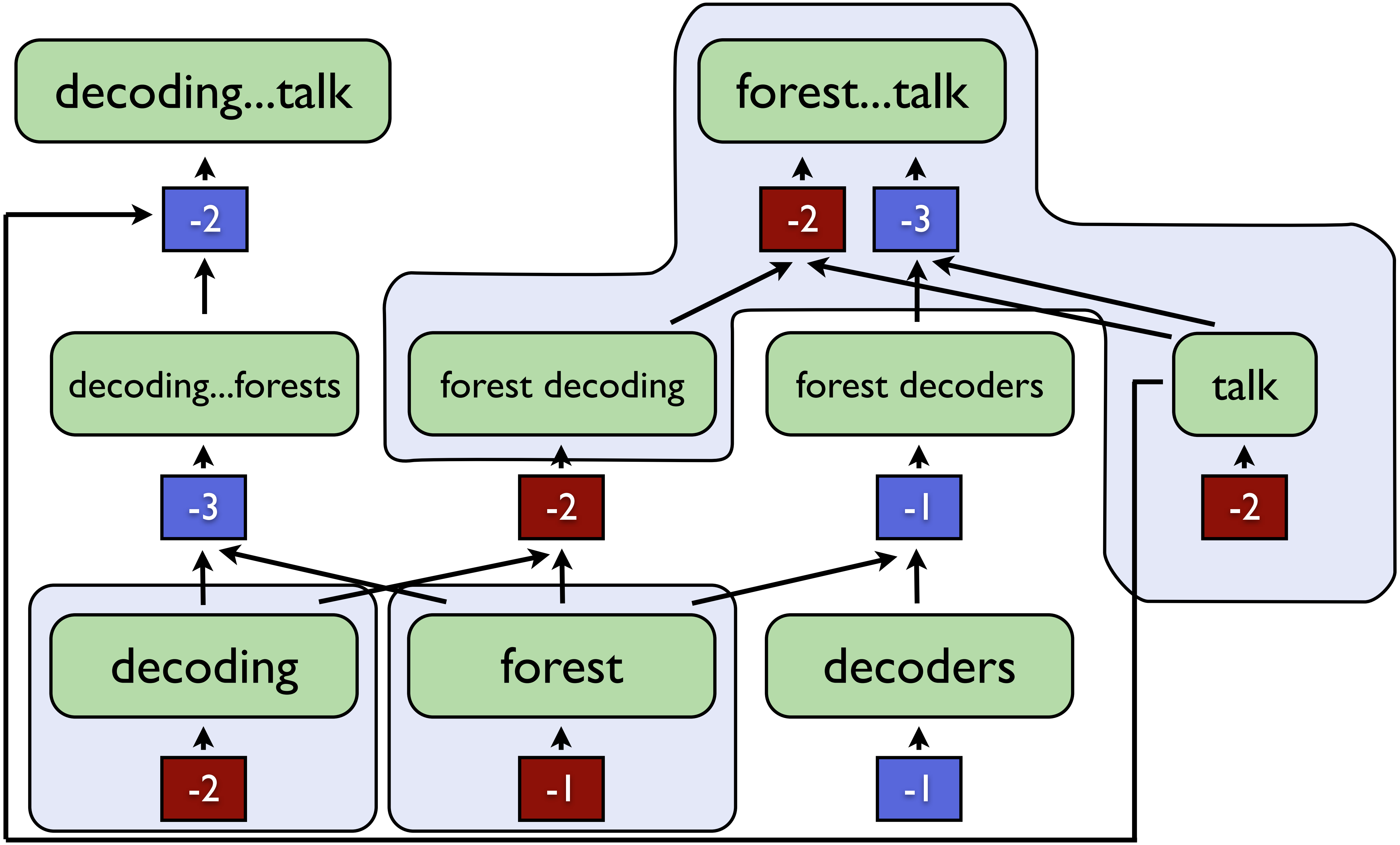
N-gram Expectations from Forests

Step 1: Compute posterior probability of each edge



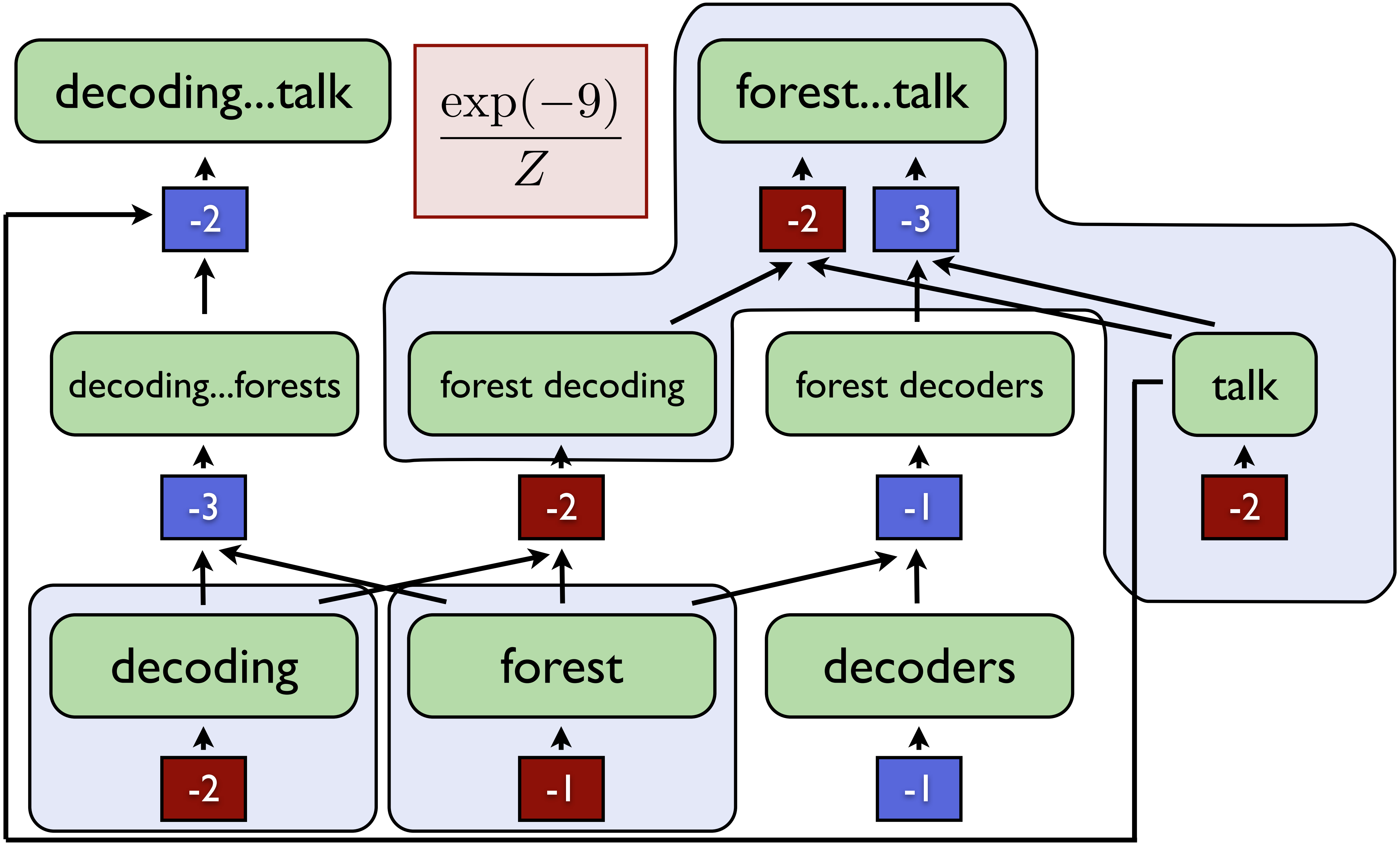
N-gram Expectations from Forests

Step 1: Compute posterior probability of each edge



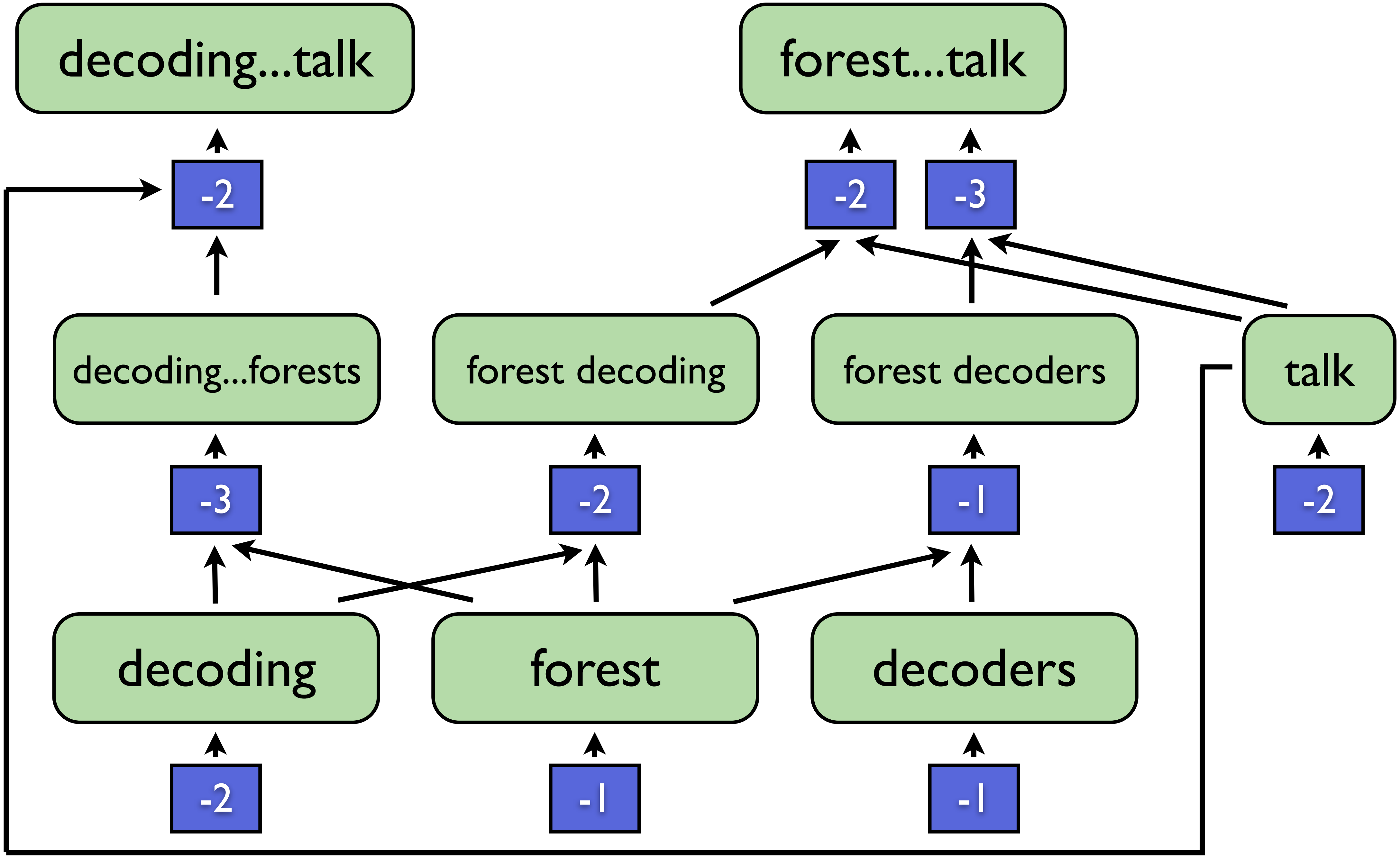
N-gram Expectations from Forests

Step 1: Compute posterior probability of each edge



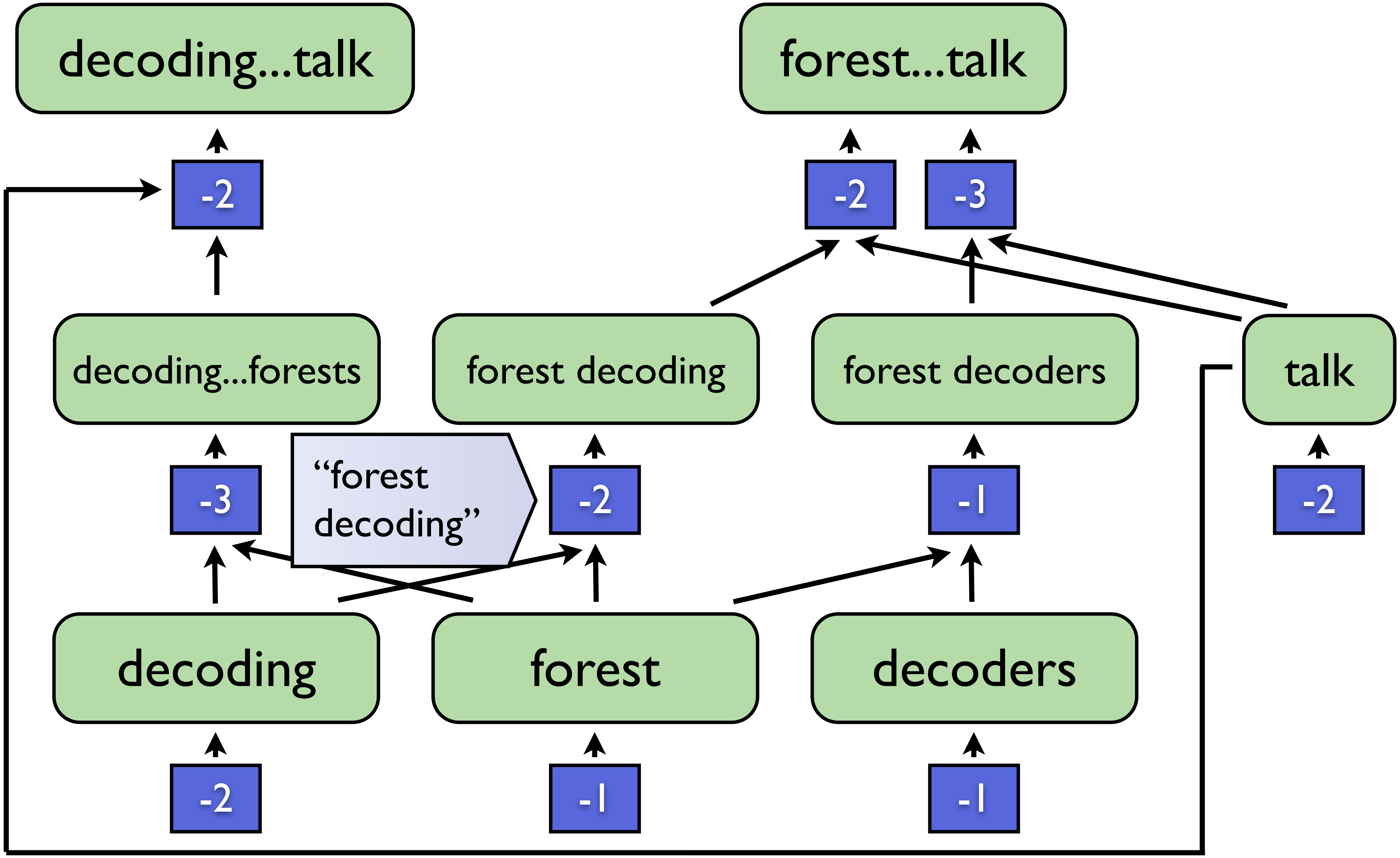
N-gram Expectations from Forests

Step 2: Find n-grams introduced by each edge



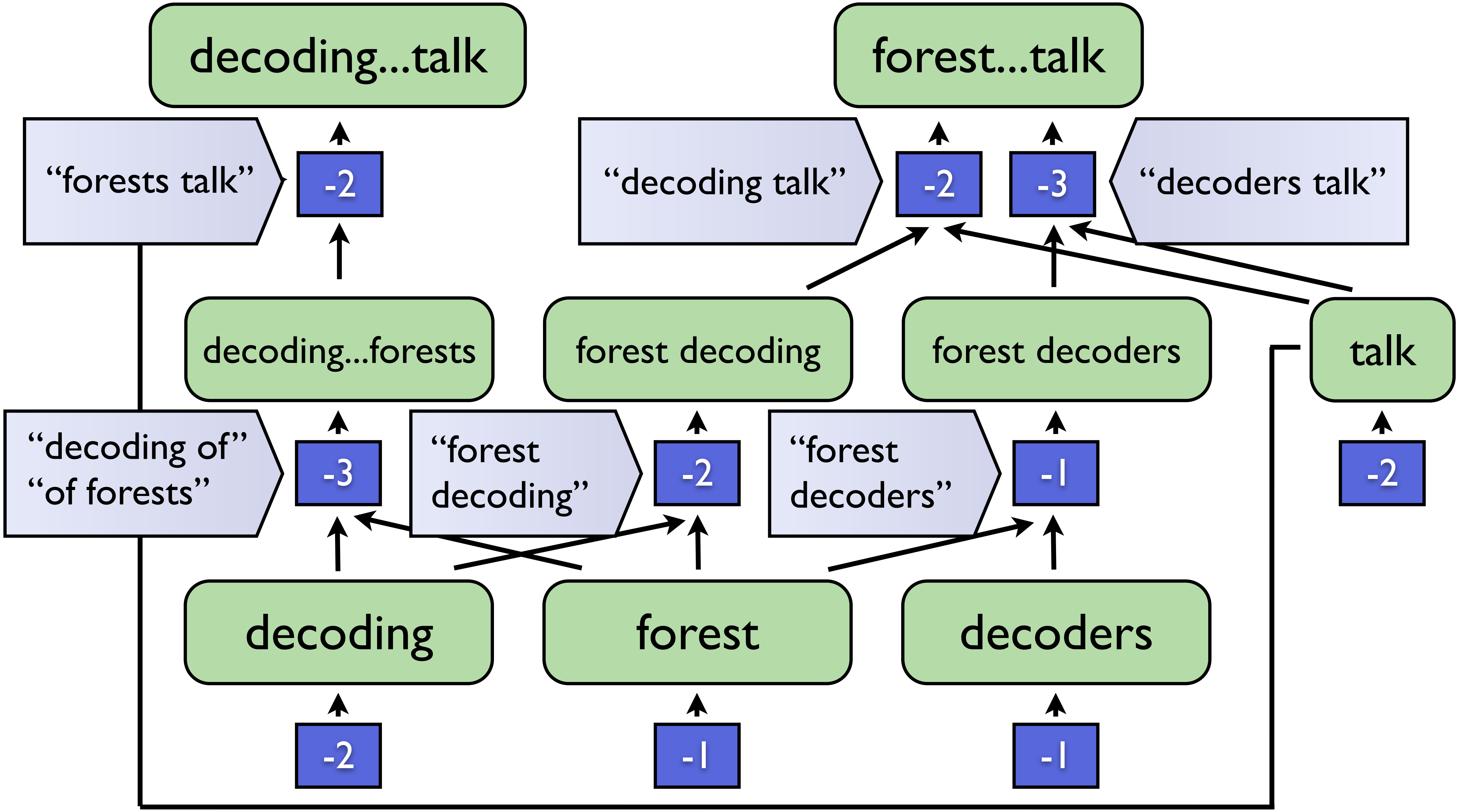
N-gram Expectations from Forests

Step 2: Find n-grams introduced by each edge



N-gram Expectations from Forests

Step 2: Find n-grams introduced by each edge



Forest-Based Expectations of Local Features

If features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Hyperedge posteriors give feature expectations

$$\mathbb{E}\left[\phi(e')\right] = \sum_{h} P(h|f) \cdot \phi(h)$$

Forest-Based Expectations of Local Features

If features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Hyperedge posteriors give feature expectations

$$\mathbb{E}\left[\phi(e')\right] = \sum_{h} \overset{\boxed{\frac{\exp(-9)}{Z}}}{\overline{P(h|f)}} \cdot \phi(h)$$

Forest-Based Expectations of Local Features

If features are local
to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Hyperedge posteriors
give feature expectations

$$\mathbb{E}\left[\phi(e')\right] = \sum_{h} \overset{\boxed{\frac{\exp(-9)}{Z}}}{\overline{P(h|f)}} \cdot \underset{\boxed{\text{“forest decoding”}}}{\underline{\phi(h)}}$$

Fast Consensus Decoding over Forests

$$\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

Build a Forest

Compute Edge Posteriors

Extract K-Best

Find Edge N-Grams

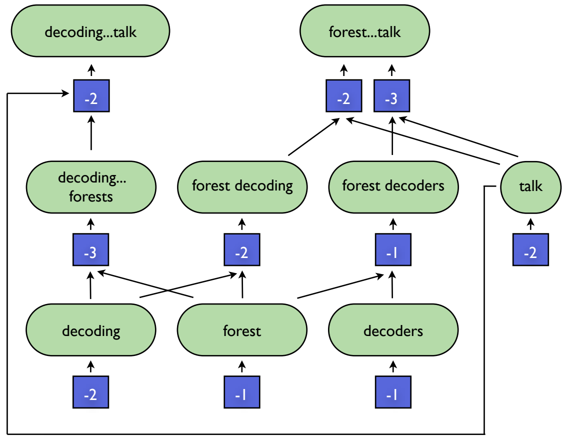
Expected N-Gram Counts

Maximize BLEU

Fast Consensus Decoding over Forests

$$\arg\max_e \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

Build a Forest



Compute Edge Posteriors

Extract K-Best

Find Edge N-Grams

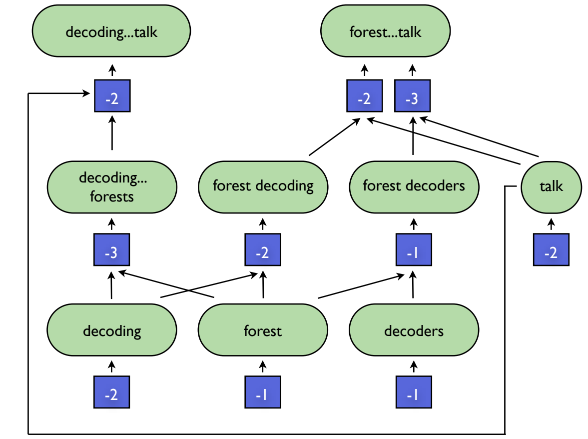
Expected N-Gram Counts

Maximize BLEU

Fast Consensus Decoding over Forests

$$\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

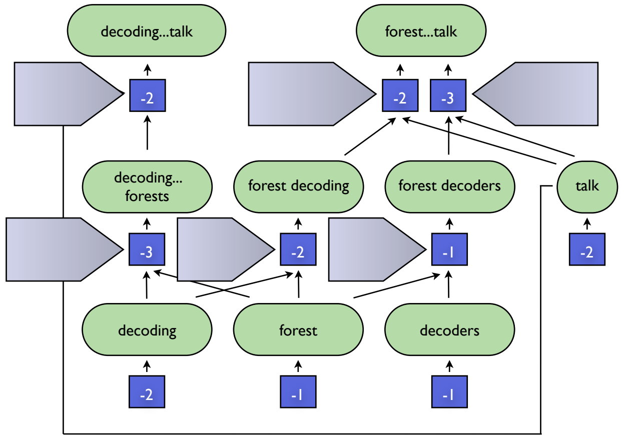
Build a Forest



Compute Edge Posteriors

Extract K-Best

Find Edge N-Grams



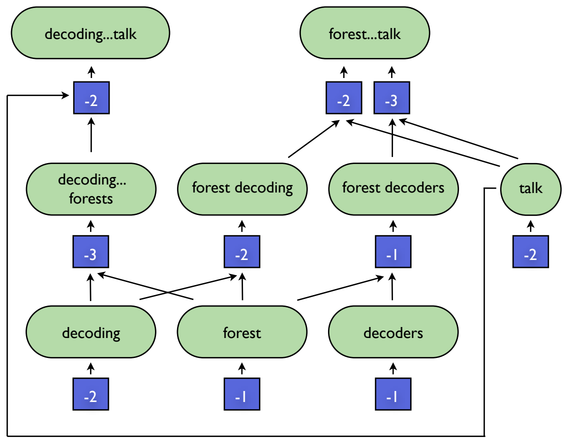
Expected N-Gram Counts

Maximize BLEU

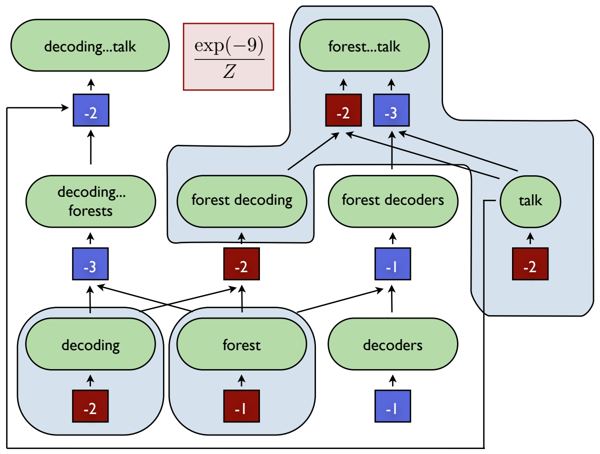
Fast Consensus Decoding over Forests

$$\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

Build a Forest

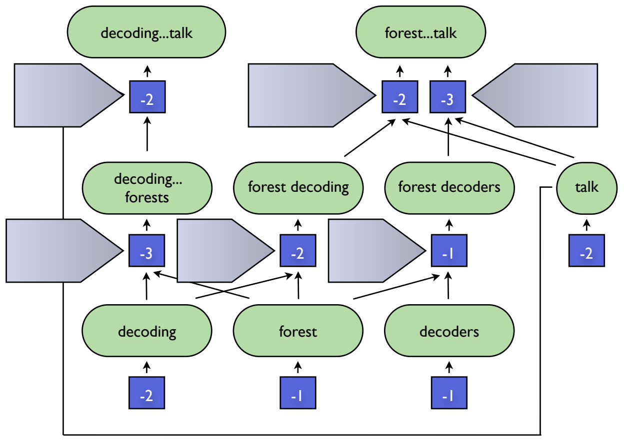


Compute Edge Posteriors



Extract K-Best

Find Edge N-Grams



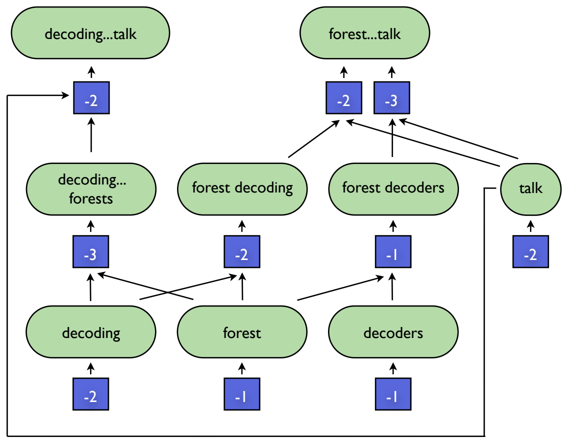
Expected N-Gram Counts

Maximize BLEU

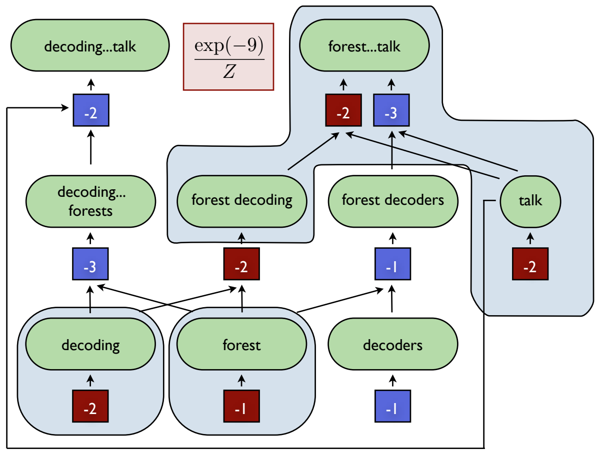
Fast Consensus Decoding over Forests

$$\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

Build a Forest

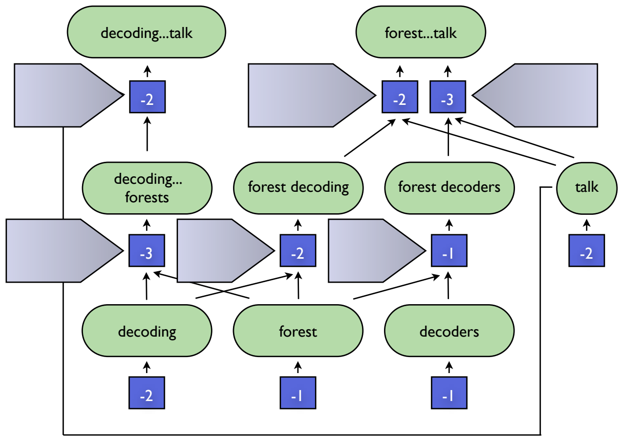


Compute Edge Posteriors



Extract K-Best

Find Edge N-Grams



Expected N-Gram Counts

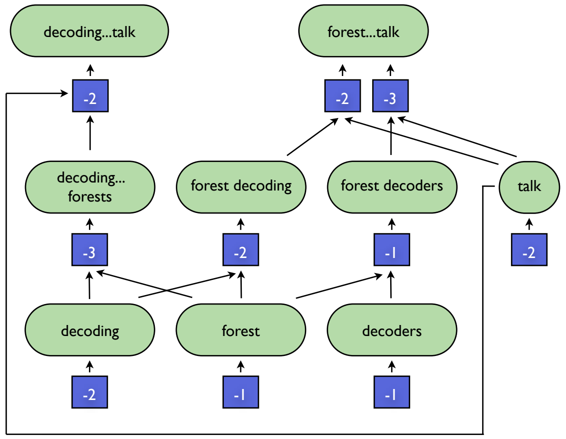
"forest decoding"	0.7
"decoding talk"	0.6
"decoders talk"	0.4
...	...

Maximize BLEU

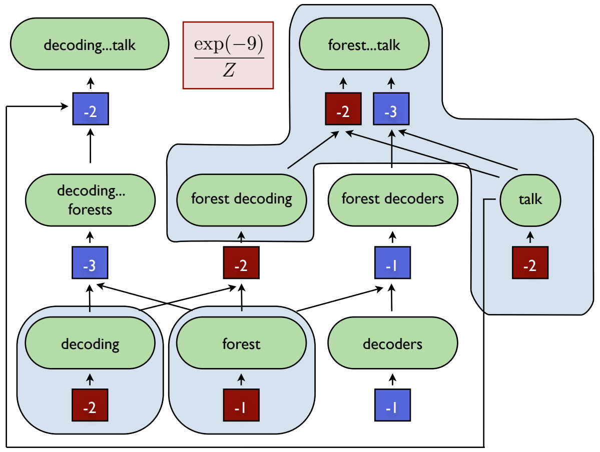
Fast Consensus Decoding over Forests

$$\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

Build a Forest



Compute Edge Posteriors



Extract K-Best

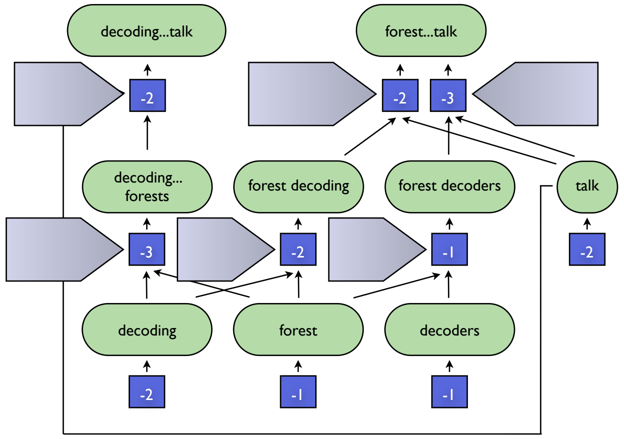
Lazy k-best extraction
[Huang and Chiang '05]

e_1: *forest decoders talk*

e_2: *forest decoding talk*

…

Find Edge N-Grams



Expected N-Gram Counts

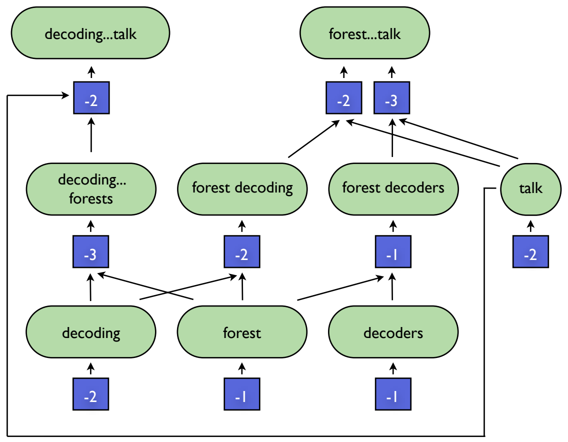
"forest decoding"	0.7
"decoding talk"	0.6
"decoders talk"	0.4
…	…

Maximize BLEU

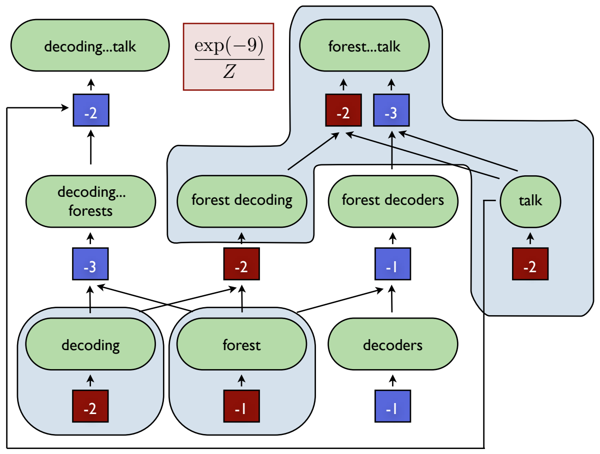
Fast Consensus Decoding over Forests

$$\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$$

Build a Forest



Compute Edge Posteriors



Extract K-Best

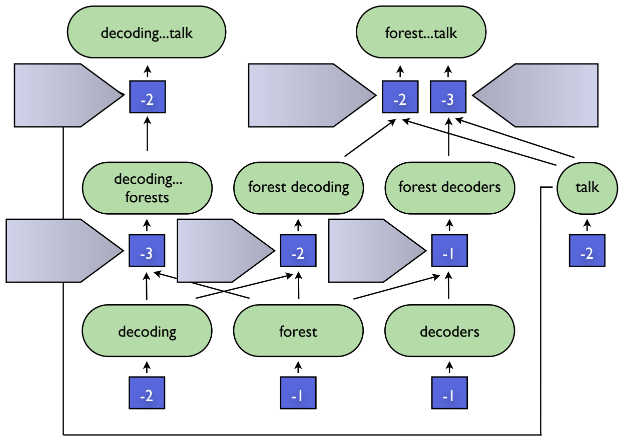
Lazy k-best extraction
[Huang and Chiang ’05]

e_1: *forest decoders talk*

e_2: *forest decoding talk*

...

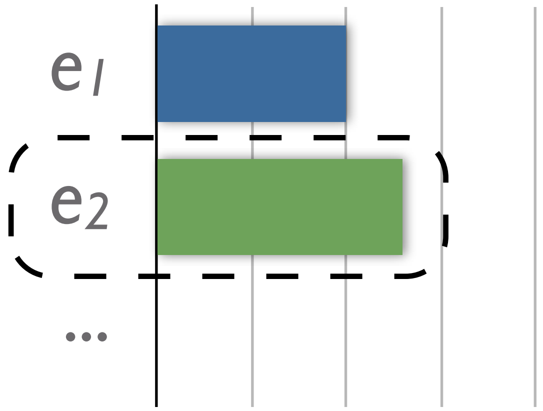
Find Edge N-Grams



Expected N-Gram Counts

“forest decoding”	0.7
“decoding talk”	0.6
“decoders talk”	0.4
...	...

Maximize BLEU



Systems Used for Experiments

Hierarchical Phrase-Based Translation (Hiero)

- Hiero rules and decoding [Chiang, ’05]
- MIRA tuning with standard, syntactic, and fine-grained distortion features [Chiang et al., ’08]

Syntax-Based Machine Translation (SBMT)

- Tree-transducer rules with no limit on non-terminal count
- Rules extracted via a variety of procedures [Galley et al., ’06; Marcu et al., ’06; DeNeefe et al., ’07]
- Tuning via MERT (Arabic-English) and MIRA (Chinese-English)

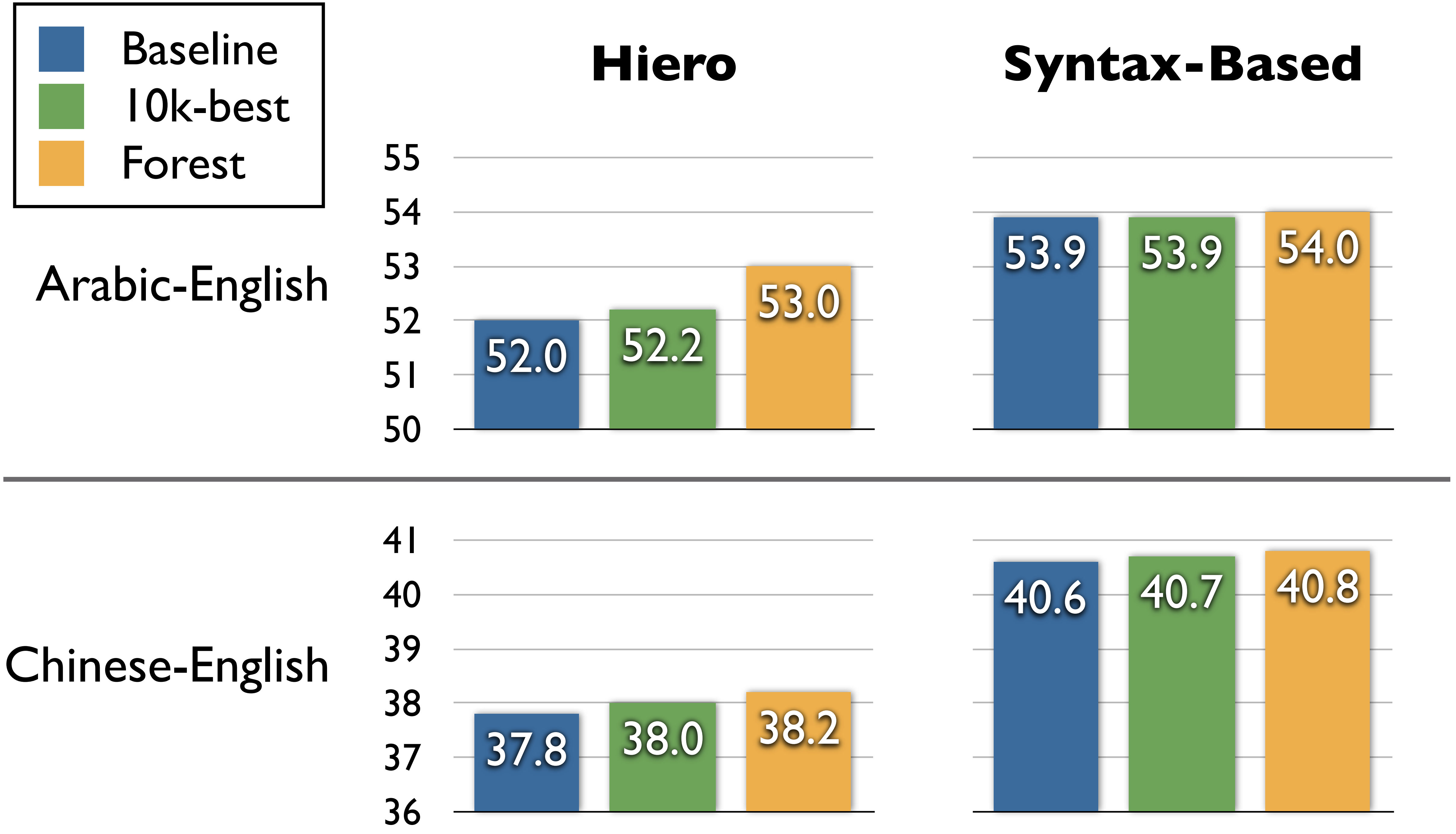
Data Conditions

Tuning and test sets drawn from NIST 2004 and 2005

	Hiero	**Syntax-Based**
Arabic-English	• 220 million word bitext • 2 billion word language model	• 220 million word bitext • 2 billion word language model
Chinese-English	• 260 million word bitext • 2 billion word language model	• 65 million word bitext • 1 billion word language model

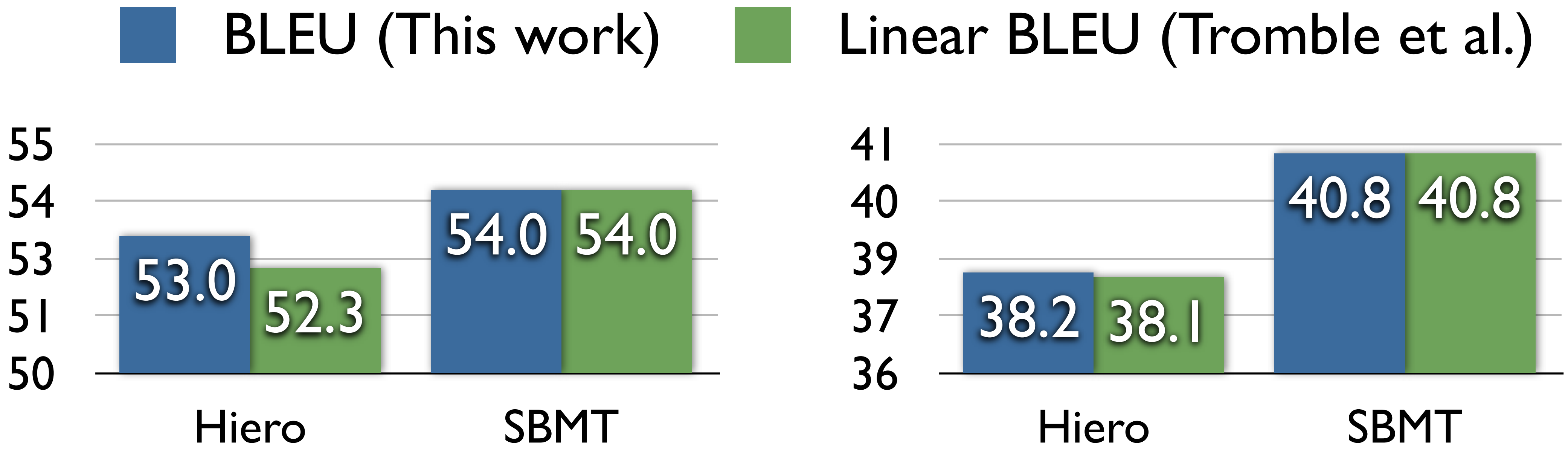
Fast Consensus Decoding Results

All results use BLEU as a similarity function



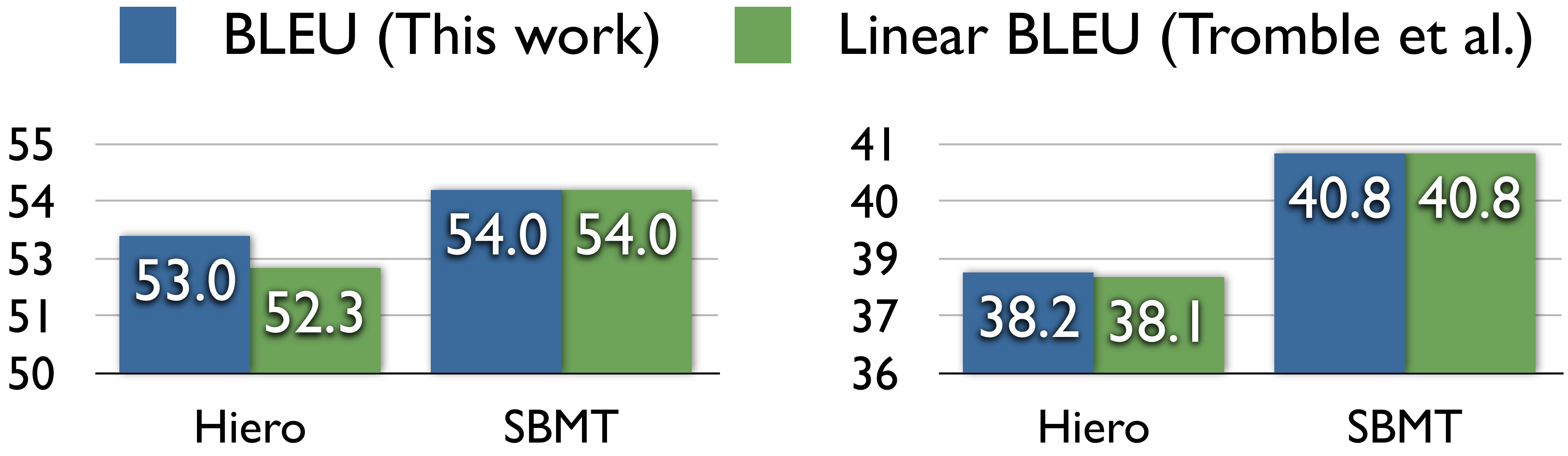
Relationship to Recent & Concurrent Work

- Tromble et al., EMNLP ’08
  - Linear approximation to BLEU for lattice MBR



Relationship to Recent & Concurrent Work

- Tromble et al., EMNLP ’08
  - Linear approximation to BLEU for lattice MBR



- Kumar et al., ACL ’09
  - Improved linear approx. to BLEU and over forests
- Li et al., ACL ’09
  - Linear objective over forests; different motivation

Training for Consensus Decoding

Decoding objective: $\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$

The model score's role is to compute n-gram expectations

Training for Consensus Decoding

Decoding objective: $\arg\max_{e} \mathrm{BLEU}(e; \mathbb{E}[\phi(e')])$

The model score's role is to compute n-gram expectations

Max-BLEU (MERT):
Maximize BLEU of the model-best derivation

CoBLEU (Gradient):
Maximize expectations of reference n-grams

Consensus Training for Consensus Decoding

Adam Pauls, John DeNero, & Dan Klein

EMNLP '09

Conclusion

- Fast consensus decoding is efficient with non-linear similarity functions
- Equivalent to MBR for linear similarity functions
- 80x speed increase over MBR with 1000-best lists (using BLEU for similarity)
- Improvements of up to 1.0 BLEU over model-best

Thanks!

Questions?

ISI
Information Sciences Institute
USC School of Engineering

University of California
CALNLP
Berkeley

Forest-Based Expectations of Local Features

Assume features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Definition of an expectation

$$\mathbb{E}\left[\phi(e')\right] = \sum_{e'} P(e'|f) \sum_{h \in e'} \phi(h)$$

Forest-Based Expectations of Local Features

Assume features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Definition of an expectation

$$\mathbb{E}\left[\phi(e')\right] = \sum_{e'} P(e'|f) \sum_{h \in e'} \phi(h)$$

Inside-outside computes hyperedge posteriors

$$P(h|f) = \sum_{e' : h \in e'} P(e'|f)$$

Forest-Based Expectations of Local Features

Assume features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Definition of an expectation

$$\mathbb{E}\left[\phi(e')\right] = \sum_{e'} P(e'|f) \sum_{h \in e'} \phi(h)$$

Inside-outside computes hyperedge posteriors

$$P(h|f) = \sum_{e' : h \in e'} P(e'|f)$$

Hyperedge posteriors give feature expectations

Forest-Based Expectations of Local Features

Assume features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Definition of an expectation

$$\mathbb{E}\left[\phi(e')\right] = \sum_{e'} P(e'|f) \sum_{h \in e'} \phi(h)$$

Inside-outside computes hyperedge posteriors

$$P(h|f) = \sum_{e' : h \in e'} P(e'|f)$$

Hyperedge posteriors give feature expectations

$$\mathbb{E}\left[\phi(e')\right] = \sum_{h} P(h|f) \cdot \phi(h)$$

Forest-Based Expectations of Local Features

Assume features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Definition of an expectation

$$\mathbb{E}\left[\phi(e')\right] = \sum_{e'} P(e'|f) \sum_{h \in e'} \phi(h)$$

Inside-outside computes hyperedge posteriors

$$P(h|f) = \sum_{e' : h \in e'} P(e'|f)$$

Hyperedge posteriors give feature expectations

$$\boxed{\frac{\exp(-9)}{Z}}$$

$$\mathbb{E}\left[\phi(e')\right] = \sum_{h} P(h|f) \cdot \phi(h)$$

Forest-Based Expectations of Local Features

Assume features are local to hyperedges

$$\phi(e') = \sum_{h \in e} \phi(h)$$

Definition of an expectation

$$\mathbb{E}\left[\phi(e')\right] = \sum_{e'} P(e'|f) \sum_{h \in e'} \phi(h)$$

Inside-outside computes hyperedge posteriors

$$P(h|f) = \sum_{e' : h \in e'} P(e'|f)$$

Hyperedge posteriors give feature expectations

$$\mathbb{E}\left[\phi(e')\right] = \sum_{h} \overset{\boxed{\frac{\exp(-9)}{Z}}}{\overline{P(h|f)}} \cdot \underset{\boxed{\text{“forest decoding”}}}{\underline{\phi(h)}}$$