Microsoft Research
微软亚洲研究院

Adaptive Multi-Compositionality for Recursive Neural Models with Applications to Sentiment Analysis

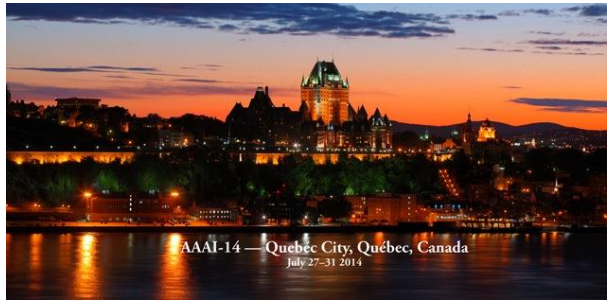
Li Dong[†*] **Furu Wei**[‡] **Ming Zhou**[‡] **Ke Xu**[†]

[†]State Key Lab of Software Development Environment, Beihang University, Beijing, China

[‡]Microsoft Research, Beijing, China

donglixp@gmail.com {fuwei,mingzhou}@microsoft.com kexu@nlsde.buaa.edu.cn

July 31, 2014

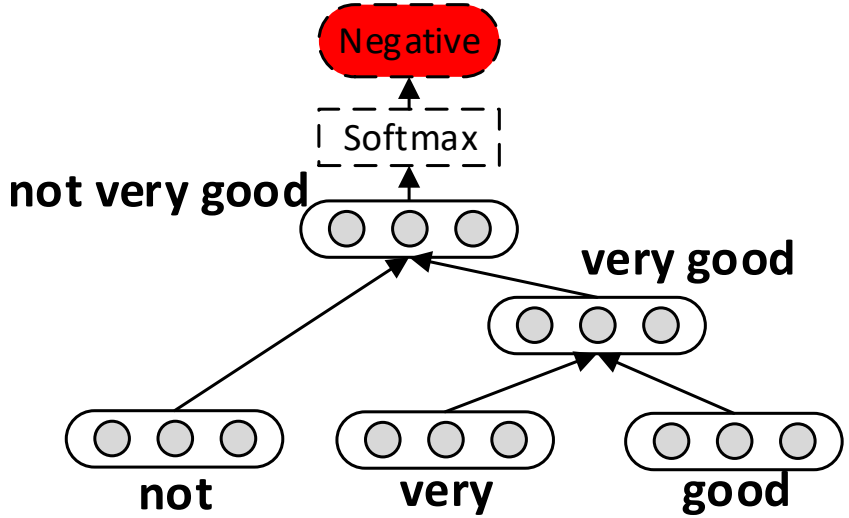


Semantic Composition

- Principle of Compositionality
  - The meaning of a complex expression is determined by the meanings of its constituent expressions and the rules used to combine them
- Compositional nature of natural language
  - Go beyond words towards sentences
- Examples
  - red car -> red + car
  - not very good -> not + (very + good)
  - eat food -> eat + food
  - …

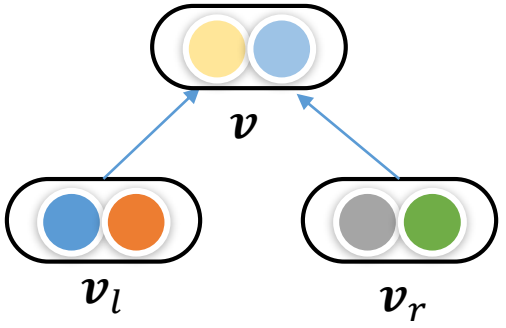
Recursive Neural Models (RNMs)

- Utilize the recursive structures of sentences to obtain the semantic representations
  - The vector representations are used as features and fed into a softmax classifier to predict their labels
- Learn to recursively perform semantic compositions in vector space
- One family of the popular deep learning models



Semantic Composition with Matrix/Tensor

- The main difference among the recursive neural models (RNMs) lies in semantic composition methods



$$v = f\left(\left[\begin{matrix} \bullet + \bullet + \bullet + \bullet \\ \bullet + \bullet + \bullet + \bullet \end{matrix}\right]\right)$$

$$\boldsymbol{v} = f\left(\boldsymbol{W}\begin{bmatrix}\boldsymbol{v}_l \\ \boldsymbol{v}_r\end{bmatrix} + \boldsymbol{b}\right)$$

RNN (Socher et al. 2011)

$$v = f\left(\left[\begin{matrix} \bullet + \bullet + \bullet + \bullet + \overbrace{\bullet + \bullet + \bullet + \bullet + \cdots + \bullet + \bullet}^{\text{intersection}} \\ \bullet + \bullet + \bullet + \bullet + \bullet + \bullet + \bullet + \bullet + \cdots + \bullet + \bullet \end{matrix}\right]\right)$$

$$\boldsymbol{v} = f\left(\begin{bmatrix}\boldsymbol{v}_l \\ \boldsymbol{v}_r\end{bmatrix}^T \boldsymbol{T}^{[1:D]} \begin{bmatrix}\boldsymbol{v}_l \\ \boldsymbol{v}_r\end{bmatrix} + \boldsymbol{W}\begin{bmatrix}\boldsymbol{v}_l \\ \boldsymbol{v}_r\end{bmatrix} + \boldsymbol{b}\right)$$

RNTN (Yu et al. 2013, Socher et al. 2013)

Problem: RNN and RNTN employ the same global composition function for all pair of input vectors

Motivation of This Work

- Use different composition functions for different types of compositions
  - Negation: not good, not bad
  - Intensification: very good, pretty bad
  - Contrast: the movie is good, but I love it
  - Sentiment word + target/aspect: good movie, low price
  - …
- Model the composition as a distribution over multiple composition functions, and adaptively select them

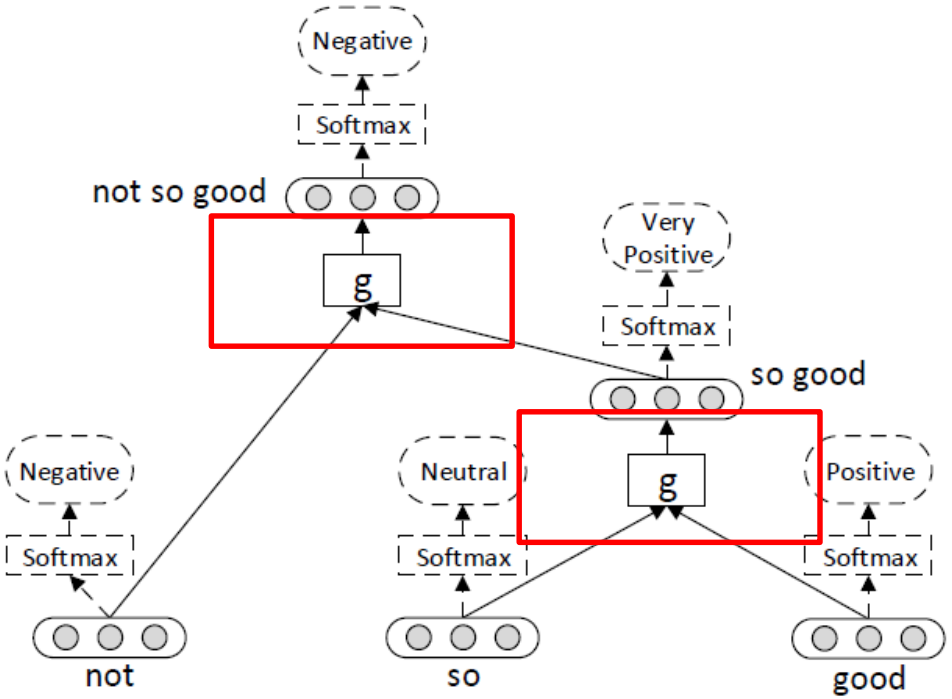


Figure 1: Composition process for "*not so good*". The g is a global composition function in recursive neural models.

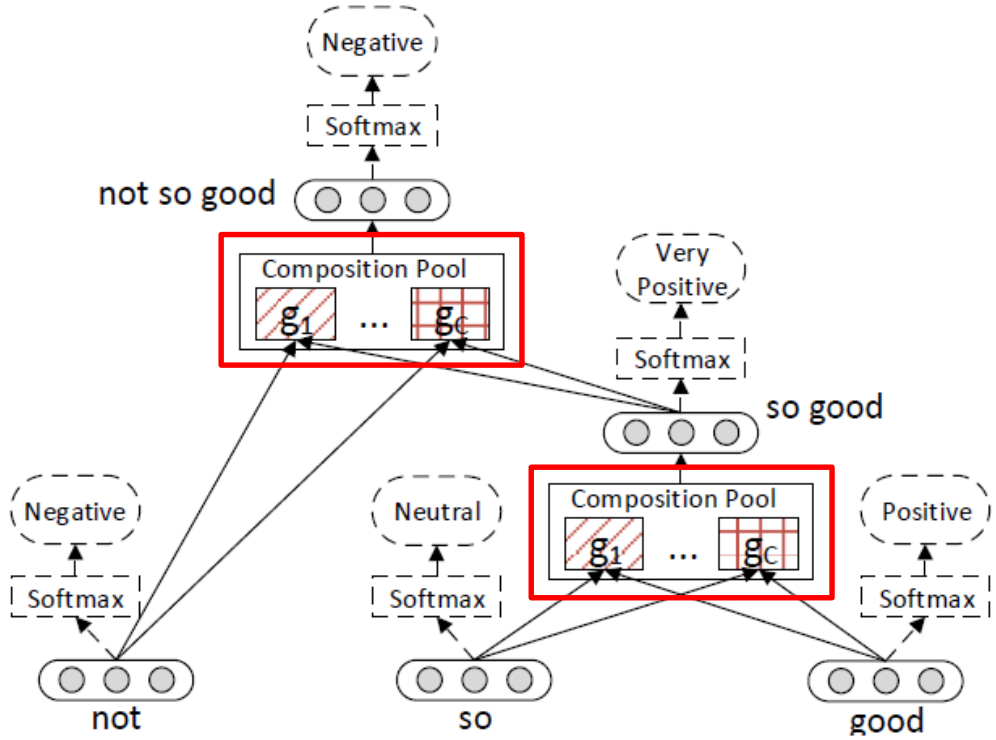


Figure 2: The composition pool consists multiple composition functions. It selects the functions depending on the input child vectors, and produces the composition result using more than one composition functions.

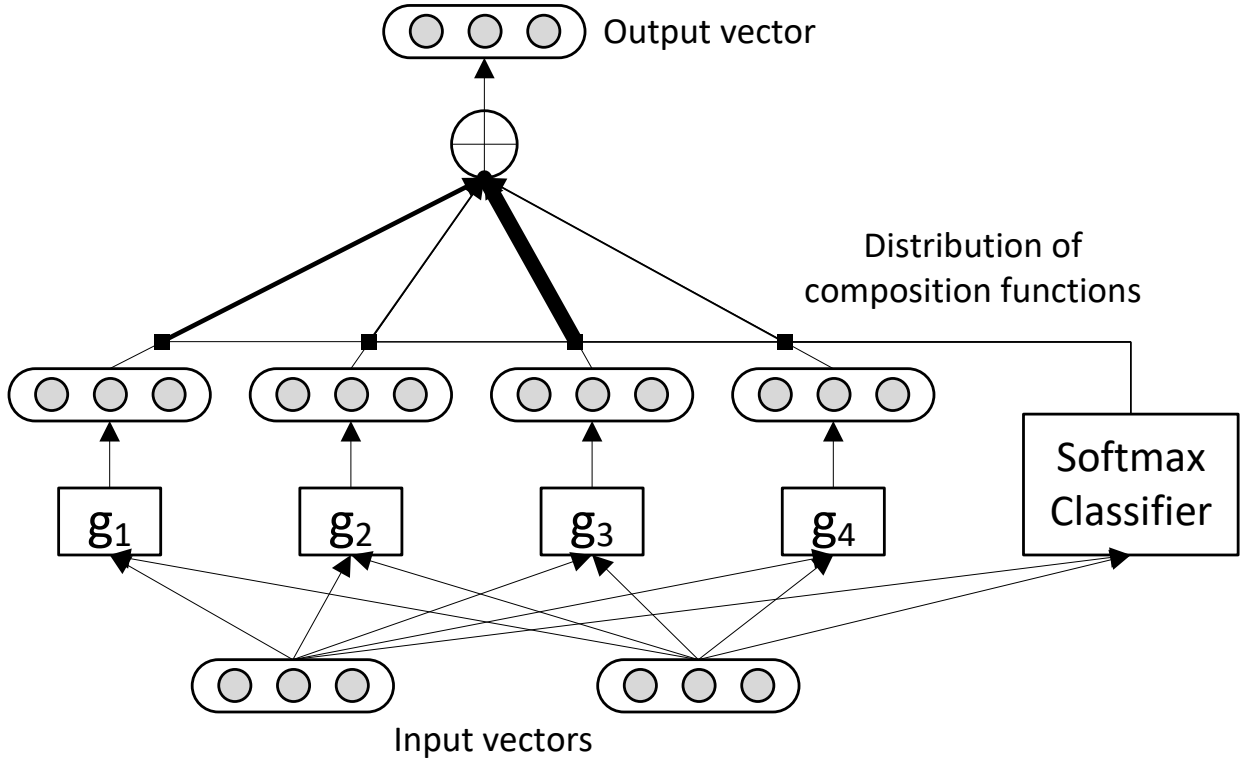
One Global Composition Function

Adaptive Multi-Compositionality

Adaptive Compositionality

- Use more than one composition functions and adaptively select them depending on the input vectors

$$\boldsymbol{v} = f\left(\sum_{h=1}^{C} P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) g_h(\boldsymbol{v}_l, \boldsymbol{v}_r)\right)$$



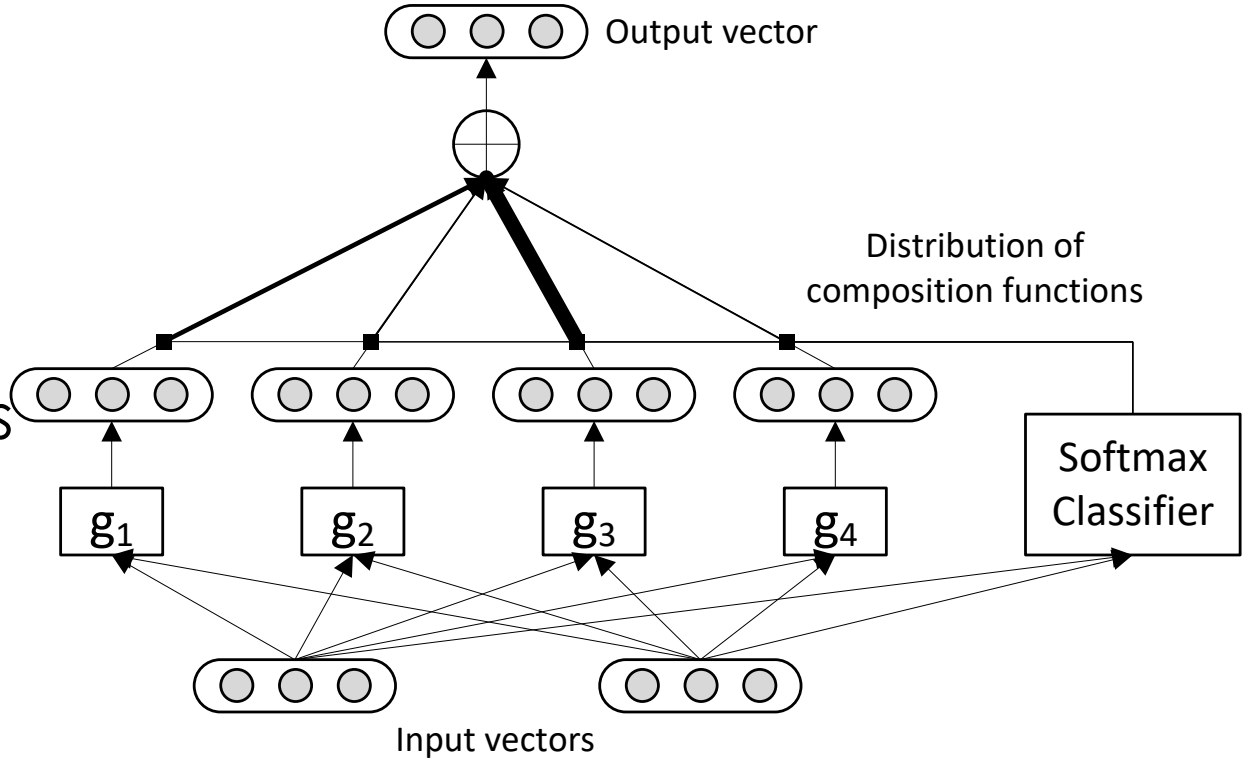
Adaptive Compositionality

- Use more than one composition functions and adaptively select them depending on the input vectors

$$\boldsymbol{v} = f\left(\sum_{h=1}^{C} P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) g_h(\boldsymbol{v}_l, \boldsymbol{v}_r)\right)$$

The h -th composition function

(Both the matrices and tensors can be used)

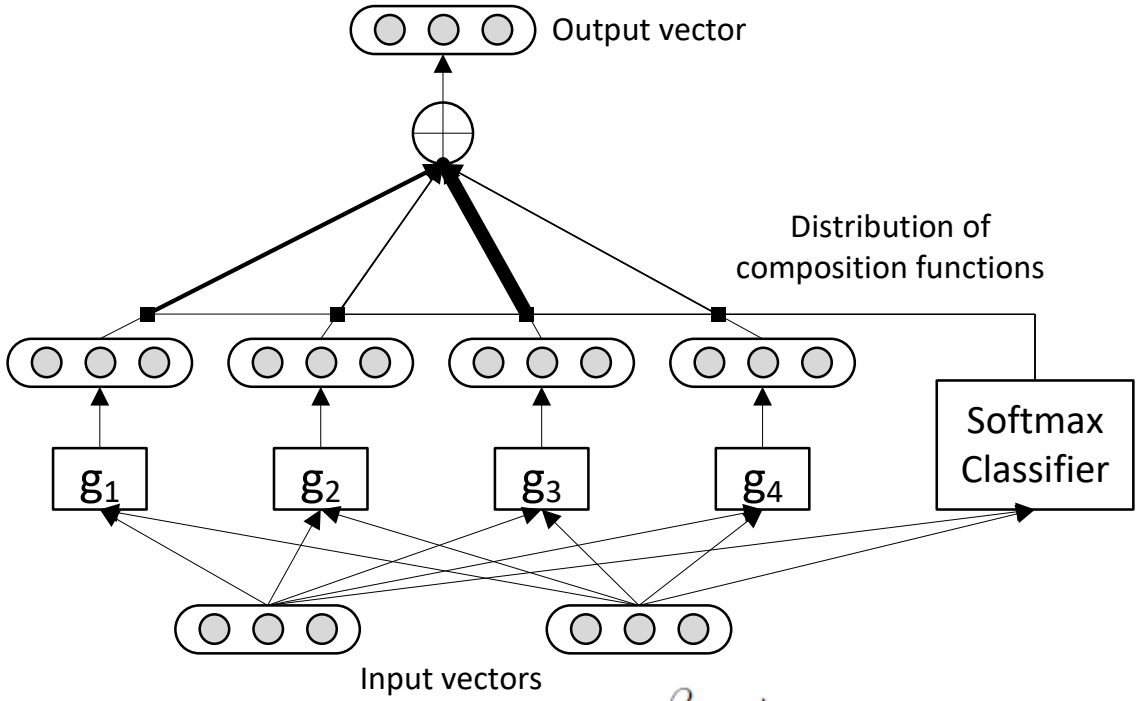


Adaptive Compositionality

- Use more than one composition functions and adaptively select them depending on the input vectors

$$\boldsymbol{v} = f\left(\sum_{h=1}^{C} P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) g_h(\boldsymbol{v}_l, \boldsymbol{v}_r)\right)$$

$$\begin{bmatrix} P(g_1|\boldsymbol{v}_l, \boldsymbol{v}_r) \\ \vdots \\ P(g_C|\boldsymbol{v}_l, \boldsymbol{v}_r) \end{bmatrix} = softmax\left(\boldsymbol{\beta S}\begin{bmatrix} \boldsymbol{v}_l \\ \boldsymbol{v}_r \end{bmatrix}\right)$$



The Boltzmann distribution is used to adaptively select g_h.

$\beta = 0$

$$P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) = \frac{1}{C}$$

Avg-AdaMC

$\beta = 1$

$$\begin{bmatrix} P(g_1|\boldsymbol{v}_l, \boldsymbol{v}_r) \\ \vdots \\ P(g_C|\boldsymbol{v}_l, \boldsymbol{v}_r) \end{bmatrix} = softmax\left(\boldsymbol{S}\begin{bmatrix} \boldsymbol{v}_l \\ \boldsymbol{v}_r \end{bmatrix}\right)$$

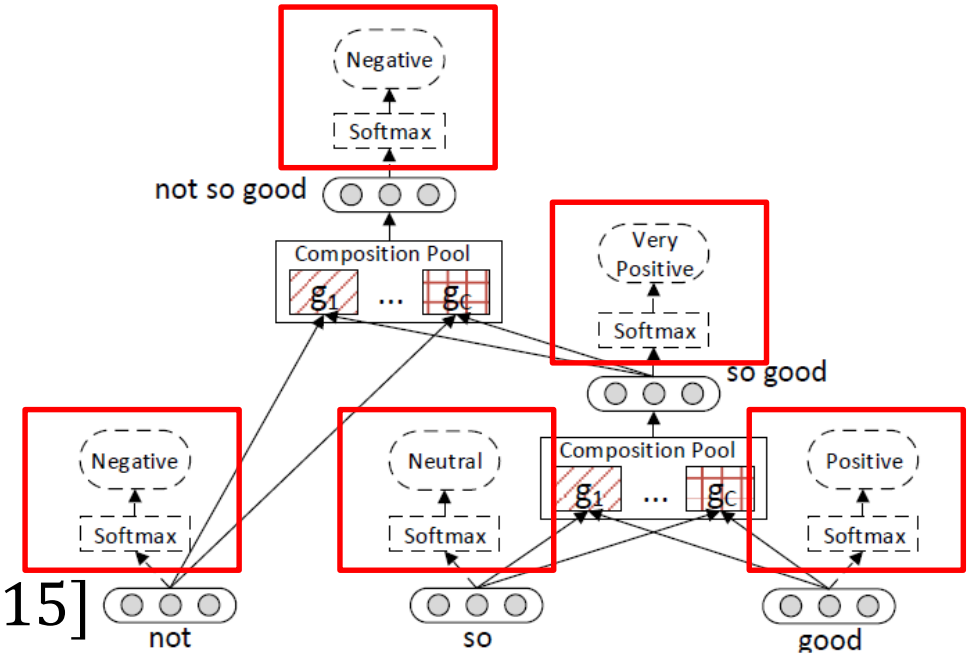
Weighted-AdaMC

$\beta \to \infty$

$$P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) = \begin{cases} 1, maxmum\ score \\ 0, otherwise \end{cases}$$

Max-AdaMC

Objective Function



- Minimize the cross-entropy error
  - Target vector $\boldsymbol{t}_j = [0 \ldots 1 \ldots 0]$
  - Predicted distribution $\boldsymbol{y}_j = [0.07 \ldots 0.69 \ldots 0.15]$

$$\min_{\Theta} E(\Theta) = -\sum_i \sum_j t_j^i \log y_j^i + \sum_{\theta \in \Theta} \lambda_\theta \|\theta\|_2^2$$

- AdaGrad (Duchi, Hazan, and Singer 2011)

$$\theta_t = \theta_{t-1} - \eta \frac{1}{\sqrt{G_t}} \frac{\partial E}{\partial \theta}\bigg|_{\theta=\theta_{t-1}}$$

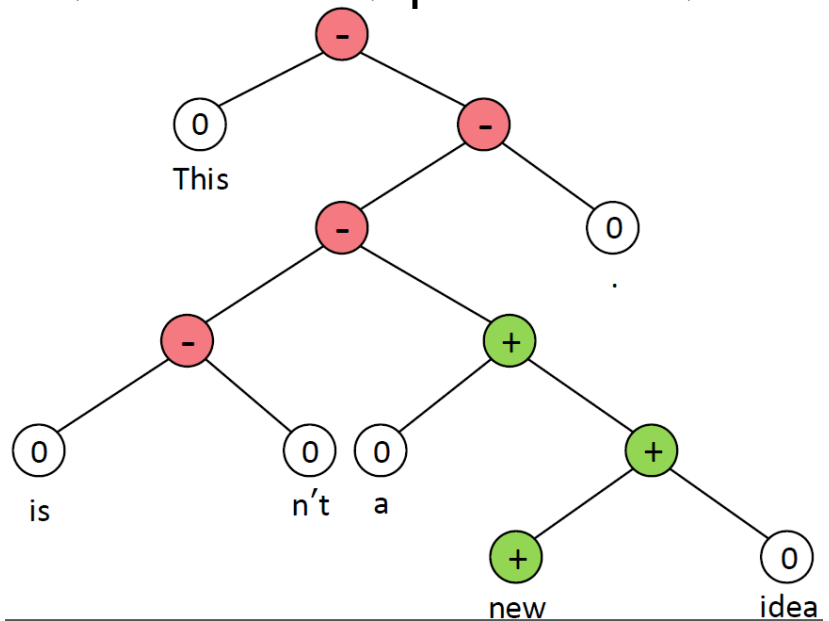
$$G_t = G_{t-1} + \left(\frac{\partial E}{\partial \theta}\bigg|_{\theta=\theta_{t-1}} \right)^2$$

Parameter Estimation

- Back-propagation algorithm: $$\boldsymbol{\delta}_m^{i\leftarrow r}=\begin{cases}\sum_k(\boldsymbol{y}_m^i-\boldsymbol{t}_m^i)\boldsymbol{U}_{mk}f'(\boldsymbol{a}_m^i), r=i\\ \sum_k \boldsymbol{\delta}_m^{par(i)\leftarrow r}\frac{\partial \boldsymbol{a}_k^{par(i)}}{\partial \boldsymbol{v}_m^i}f'(\boldsymbol{a}_m^i), r\in anc(i)\end{cases}$$
- Classification: $\frac{\partial E}{\partial \boldsymbol{U}_{mn}}=\sum_i[\boldsymbol{v}_n^i(\boldsymbol{y}_m^i-\boldsymbol{t}_m^i)]$
- Composition selection: $$\frac{\partial E}{\partial \boldsymbol{S}_{mn}}=\begin{cases}\sum_i\sum_{r\in bp(i)}\sum_k \boldsymbol{\delta}_k^{i\leftarrow r}\sum_h \boldsymbol{a}_k^{i,g_h}\boldsymbol{x}_n^i\beta P(g_h|\boldsymbol{v}_l^i,\boldsymbol{v}_r^i)(P(g_h|\boldsymbol{v}_l^i,\boldsymbol{v}_r^i)-1), h=m\\ \sum_i\sum_{r\in bp(i)}\sum_k \boldsymbol{\delta}_k^{i\leftarrow r}\sum_h \boldsymbol{a}_k^{i,g_h}\boldsymbol{x}_n^i\beta P(g_h|\boldsymbol{v}_l^i,\boldsymbol{v}_r^i)P(g_m|\boldsymbol{v}_l^i,\boldsymbol{v}_r^i), h\neq m\end{cases}$$
- Linear composition: $\frac{\partial E}{\partial \boldsymbol{W}_{mn}}=\sum_i\sum_{r\in bp(i)}\boldsymbol{\delta}_m^{i\leftarrow r}\boldsymbol{x}_n^i P(g_h|\boldsymbol{v}_l^i,\boldsymbol{v}_r^i)$
- Tensor Composition: $\frac{\partial E}{\partial \boldsymbol{V}^{h[d]}_{mn}}=\sum_i\sum_{r\in bp(i)}\boldsymbol{\delta}_d^{i\leftarrow r}\boldsymbol{x}_{\mathrm{m}}^i\boldsymbol{x}_n^i P(g_h|\boldsymbol{v}_l^i,\boldsymbol{v}_r^i)$
- Word Embedding: $\frac{\partial E}{\partial \boldsymbol{L}_d^w}=\sum_{[i]=w}\sum_{r\in bp(i)}\boldsymbol{\delta}_d^{i\leftarrow r}$

Stanford Sentiment Treebank

- 10,662 critic reviews in Rotten Tomatoes
- 215,154 phrases from results of Stanford Parser
- The workers in Amazon Mechanical Turk annotate polarity levels for all these phrases
- The sentiment scales are merged to five categories (very negative, negative, neutral, positive, very positive)



Method	Fine-grained	Pos./Neg.
SVM	40.7	79.4
MNB	41.0	81.8
bi-MNB	41.9	83.1
VecAvg	32.7	80.1
MV-RNN	44.4	82.9
RNN	43.2	82.4
Avg-AdaMC-RNN	43.4	84.9
Max-AdaMC-RNN	43.8	85.6
Weighted-AdaMC-RNN	45.4	86.5
AdaMC-RNN	**45.8**	**87.1**
RNTN	45.7	85.4
Avg-AdaMC-RNTN	45.7	86.3
Max-AdaMC-RNTN	45.6	86.6
Weighted-AdaMC-RNTN	**46.3**	**88.4**
AdaMC-RNTN	**46.7**	**88.5**

Results of evaluation on the Sentiment Treebank. The top three methods are in bold. Our methods achieve best performances when \beta is set to 2.

$$\boldsymbol{v} = f\left(\sum_{h=1}^{C} P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) g_h(\boldsymbol{v}_l, \boldsymbol{v}_r)\right)$$

$$\begin{bmatrix} P(g_1|\boldsymbol{v}_l, \boldsymbol{v}_r) \\ \vdots \\ P(g_C|\boldsymbol{v}_l, \boldsymbol{v}_r) \end{bmatrix} = softmax\left(\boldsymbol{\beta S} \begin{bmatrix} \boldsymbol{v}_l \\ \boldsymbol{v}_r \end{bmatrix}\right)$$

Avg-AdaMC

$\beta = 0$

$$P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) = \frac{1}{C}$$

Weighted-AdaMC

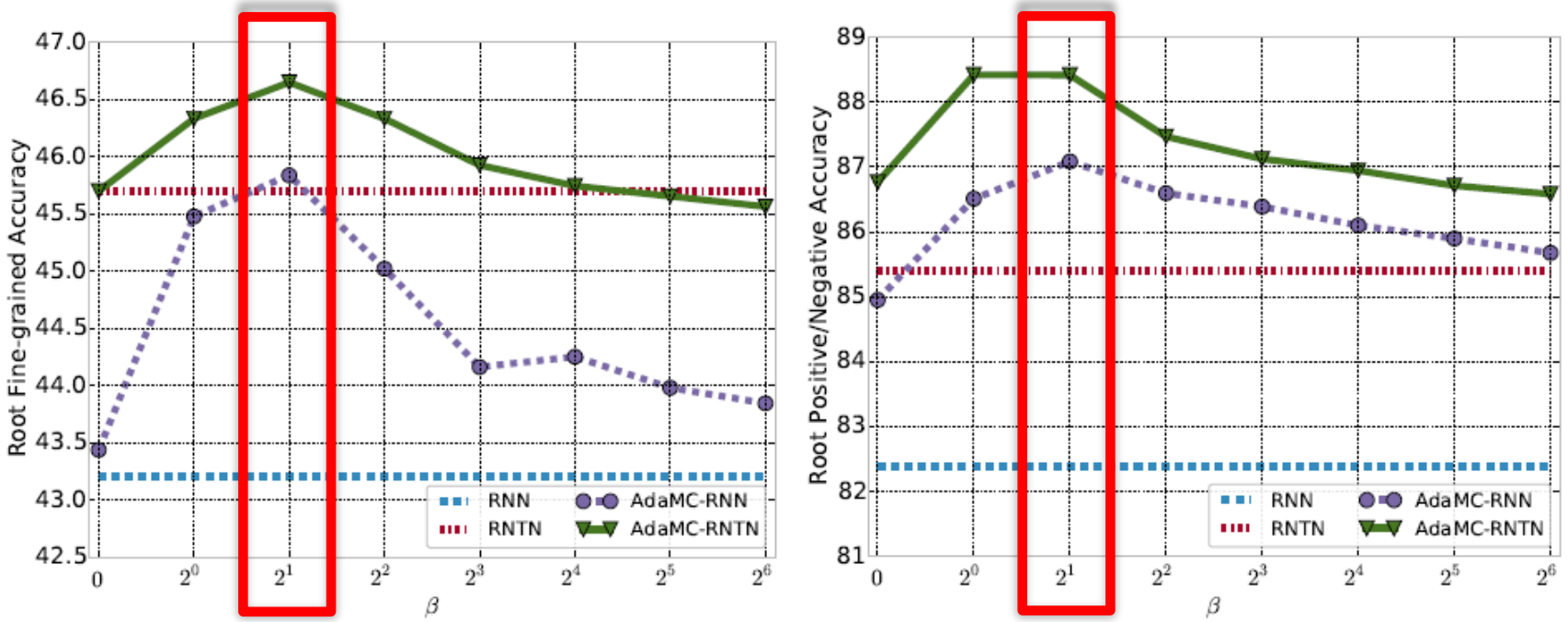
$\beta = 1$

$$\begin{bmatrix} P(g_1|\boldsymbol{v}_l, \boldsymbol{v}_r) \\ \vdots \\ P(g_C|\boldsymbol{v}_l, \boldsymbol{v}_r) \end{bmatrix} = softmax\left(\boldsymbol{S} \begin{bmatrix} \boldsymbol{v}_l \\ \boldsymbol{v}_r \end{bmatrix}\right)$$

Max-AdaMC

$\beta \to \infty$

$$P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) = \begin{cases} 1, max\ score \\ 0, otherwise \end{cases}$$



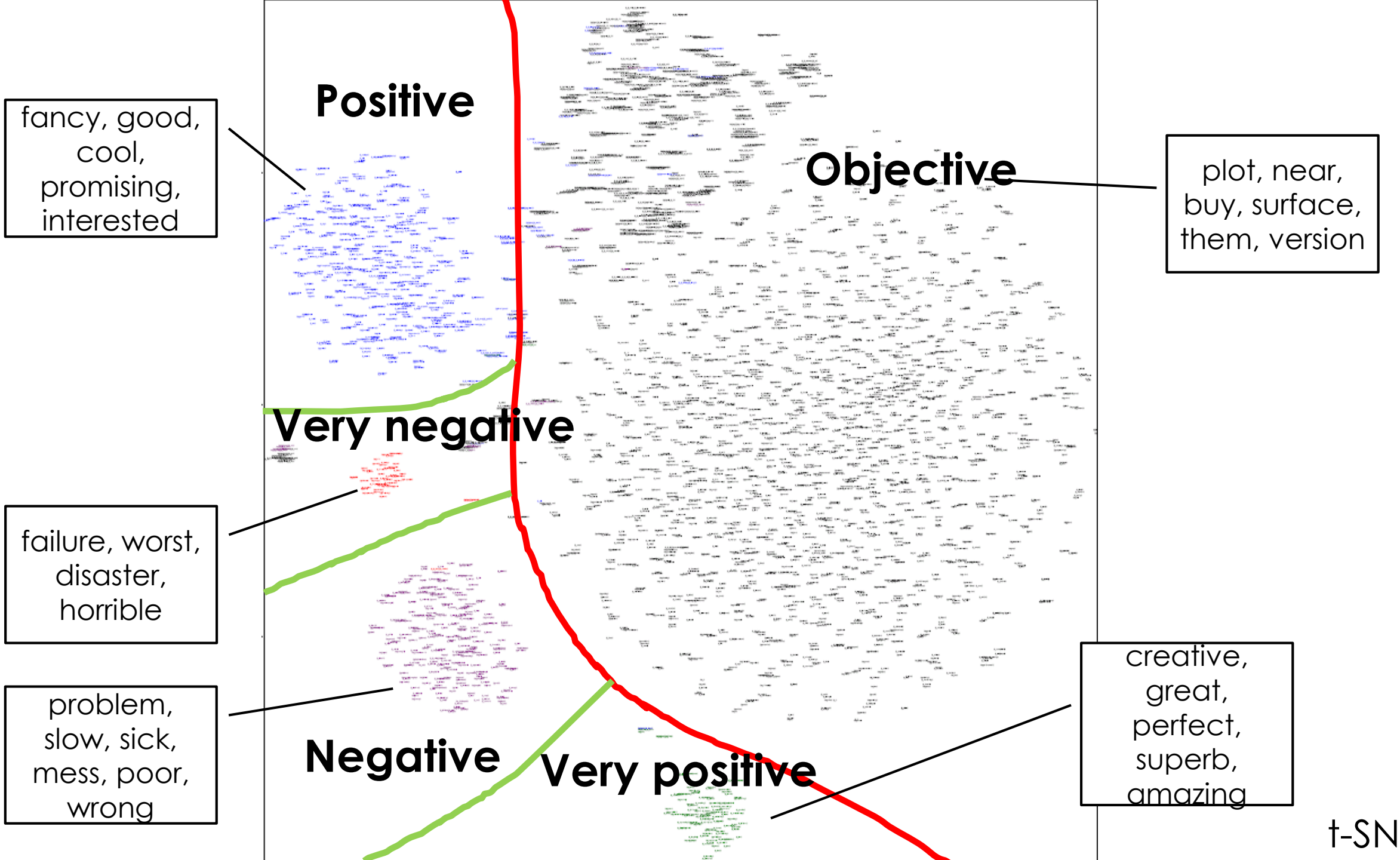
(a) Root Fine-grained Accuracy

(b) Root Pos/Neg Accuracy

Figure 3: The curve shows the accuracy for root nodes as $\beta = 0, 2^0, 2^1, \ldots, 2^6$ increases. AdaMC-RNN and AdaMC-RNTN achieve the best results at $\beta = 2^1$.

Vector Representations

Word/Phrase	Neighboring Words/Phrases in the Vector Space
good	cool, fantasy, classic, watchable, attractive
boring	dull, bad, disappointing, horrible, annoying
ingenious	extraordinary, inspirational, imaginative, thoughtful, creative
soundtrack	execution, animation, cast, colors, scene
good actors	good ideas, good acting, good looks, good sense, great cast
thought-provoking film	beautiful film, engaging film, lovely film, remarkable film, riveting story
painfully bad	how bad, too bad, really bad, so bad, very bad
not a good movie	isn't much fun, isn't very funny, nothing new, isn't as funny of clichés


Positive
fancy, good, cool, promising, interested
Objective
plot, near, buy, surface, them, version
Very negative
failure, worst, disaster, horrible
problem, slow, sick, mess, poor, wrong
Negative
Very positive
creative, great, perfect, superb, amazing
t-SNE

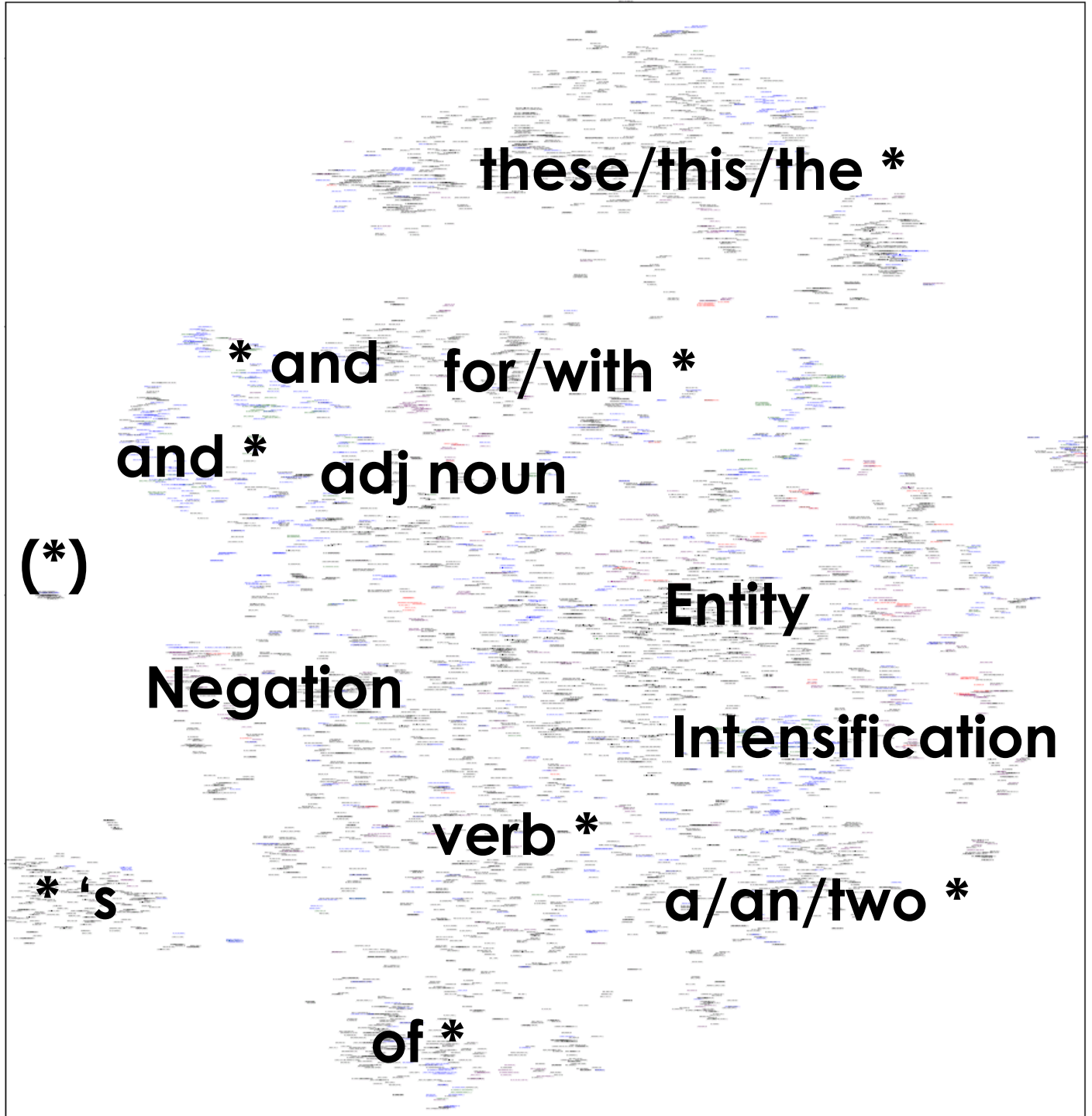
Composition Pairs in the Composition Space

- For the composition pair $(\boldsymbol{v}_l, \boldsymbol{v}_r)$, we use the distribution of the composition functions $\begin{bmatrix} P(g_1|\boldsymbol{v}_l, \boldsymbol{v}_r) \\ \vdots \\ P(g_C|\boldsymbol{v}_l, \boldsymbol{v}_r) \end{bmatrix}$ to query its neighboring pairs

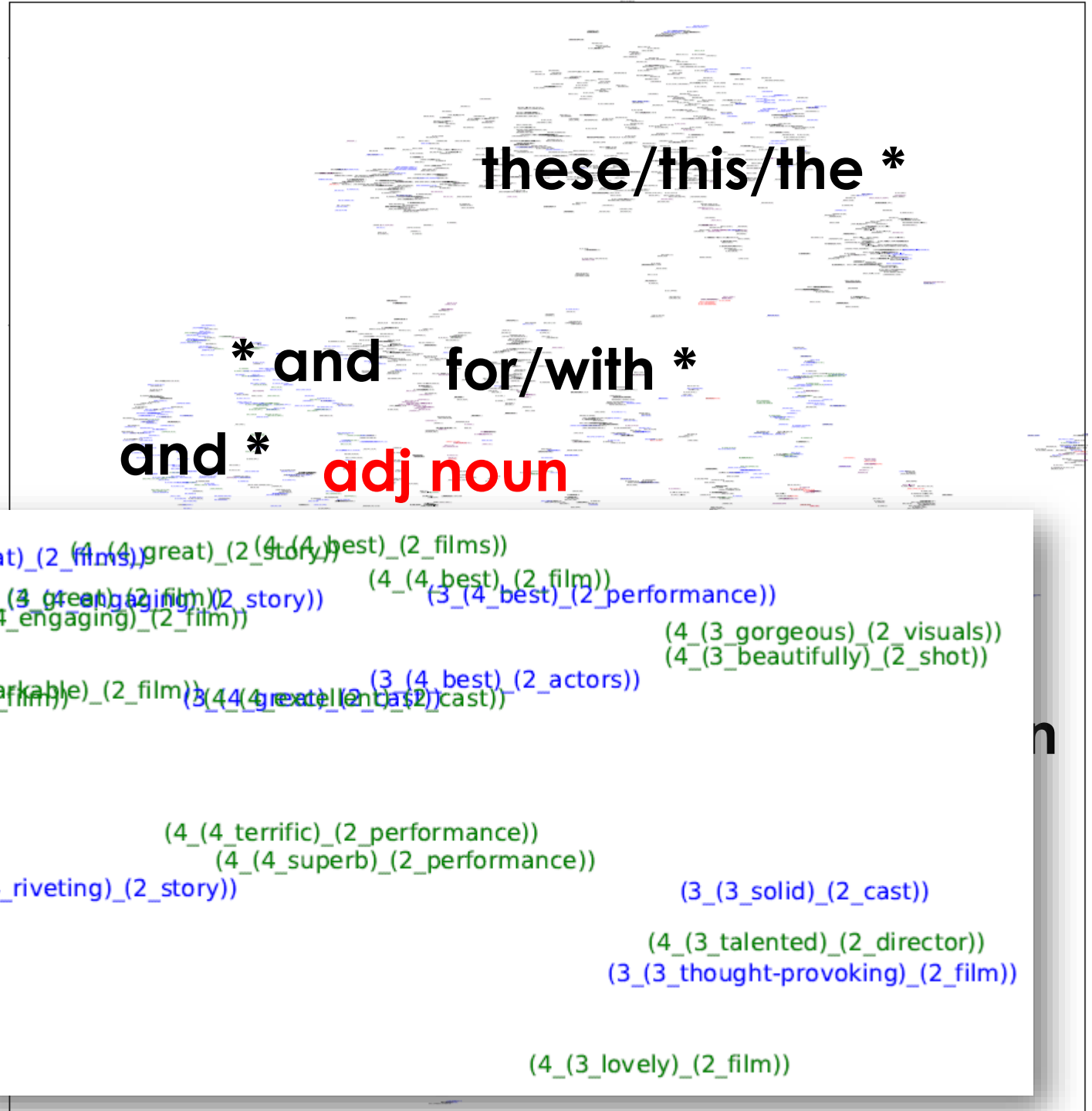
Composition Pair	Neighboring Composition Pairs
really bad	very bad / only dull / much bad / extremely bad / (all that) bad
(is n't) (necessarily bad)	(is n't) (painfully bad) / not mean-spirited / not (too slow) / not well-acted / (have otherwise) (been bland)
great (Broadway play)	great (cinematic innovation) / great subject / great performance / energetic entertainment / great (comedy filmmaker)
(arty and) jazzy	(Smart and) fun / (verve and) fun / (unique and) entertaining / (gentle and) engrossing / (warmth and) humor

Visualization:
Composition Pairs

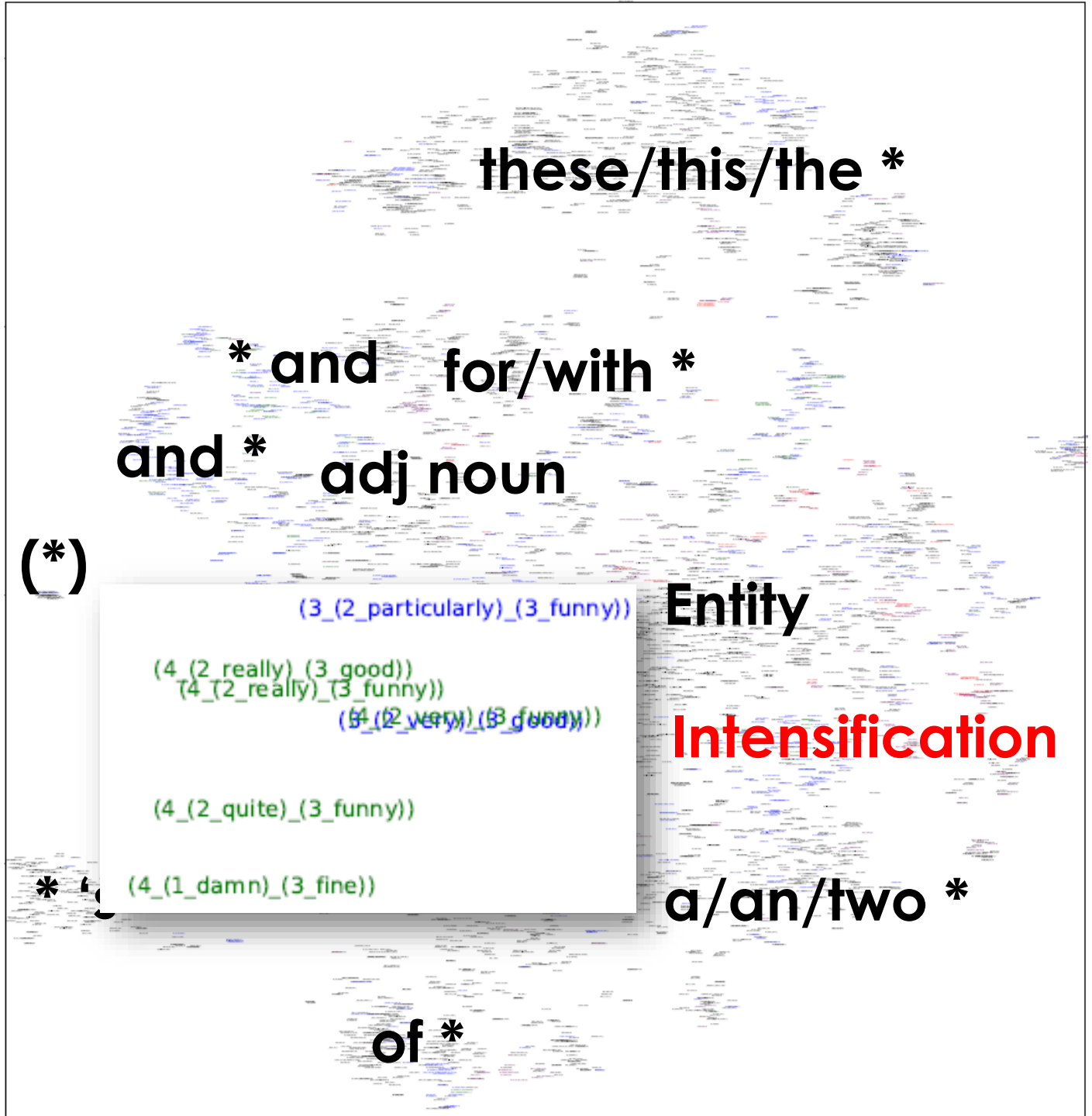
$$\begin{bmatrix} P(g_1|\boldsymbol{v}_l, \boldsymbol{v}_r) \\ \vdots \\ P(g_C|\boldsymbol{v}_l, \boldsymbol{v}_r) \end{bmatrix}$$



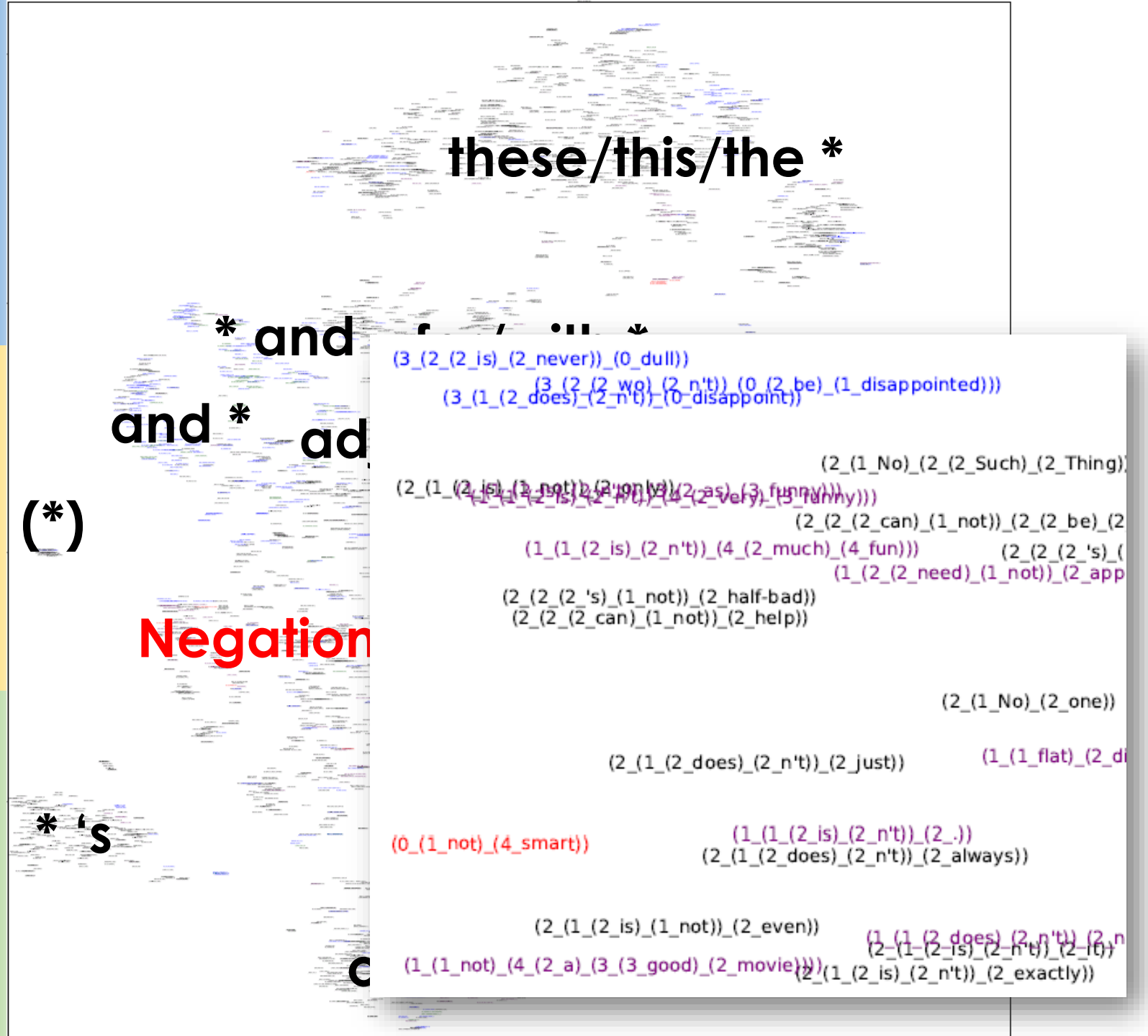
t-SNE



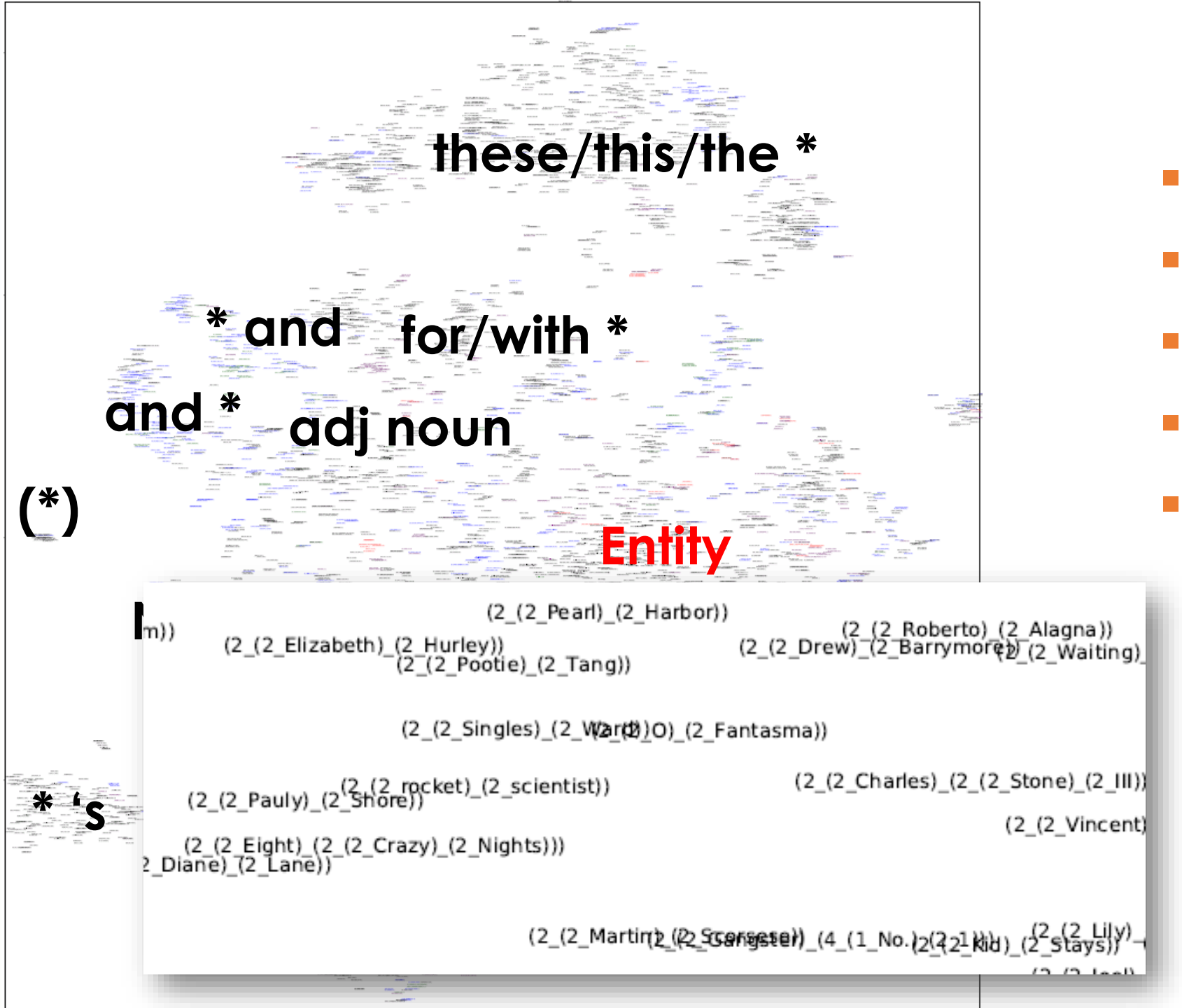
- Best films
- Riveting story
- Solid cast
- Talented director
- Gorgeous visuals



- Really good
- Quite funny
- Damn fine
- Very good
- Particularly funny



- Is never dull
- Not smart
- Not a good movie
- Is n't much fun
- Wo n't be disappointed



- Roberto Alagna
- Pearl Harbor
- Elizabeth Hurley
- Diane Lane
- Pauly Shore

Future Work

- Use AdaMC for the other NLP tasks
- Utilize external information to adaptively select the composition functions
  - Part-of-speech tags
  - Syntactic parsing results
- Mix different composition types together
  - Linear combination approach (RNN)
  - Tensor-based approach (RNTN)
  - Multiplication approach
- …

Microsoft Research
微软亚洲研究院

THANKS!