

# BEYOND WORD IMPORTANCE: CONTEXTUAL DECOMPOSITION TO EXTRACT INTERACTIONS FROM LSTMS

---

Mohammad Samavatian

This is following previous paper summary for contextual decomposition

# What? Why? How?



LSTMS have become a basic building block for NLP while remaining largely inscrutable



Aim to add interpretability without altering accuracy



Contextual Decomposition (CD) is a method for explaining individual prediction without modification of the LSTM model



Extract how words are combined to yield the final prediction

# Black Boxes Are Limited



Deep Learning models produce very accurate prediction



Predictive accuracy came at the cost of interpretability

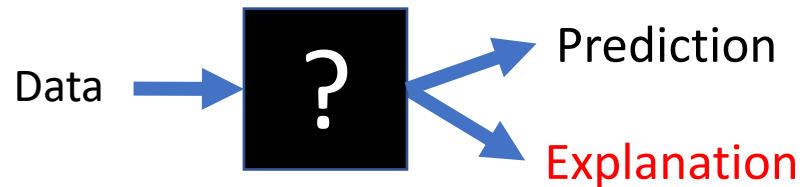


Often predictions are not enough, We want to know why a prediction was made

# What is Paper About?



**Aim to explain individual prediction made by a trained model**



**Focus on Long-Short Term Memory network (LSTM)**

Popular deep learning model for text

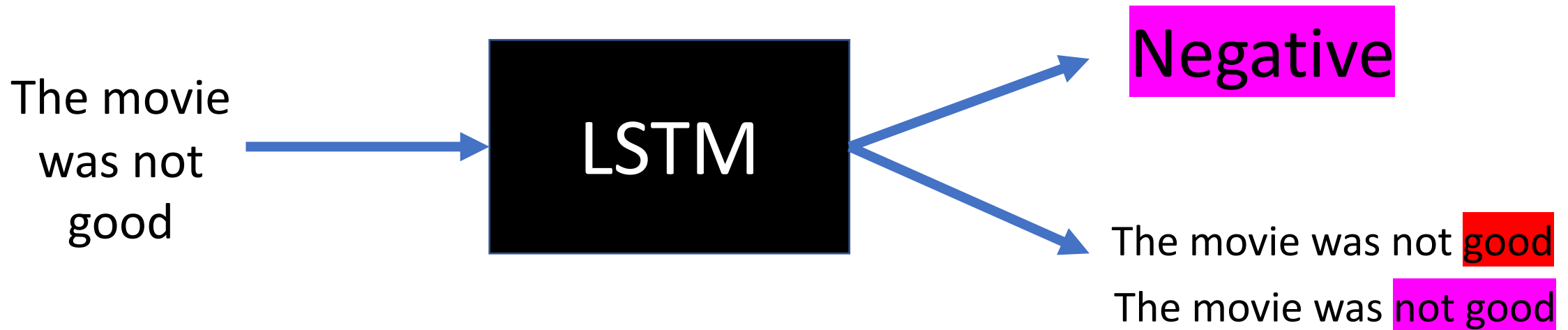
# How to Explain Prediction

- Binary sentiment analysis
  - Give movie review: Positive or Negative?
- LSTM is bag of words classifier:
  - Identify important words



# More Complicated Example

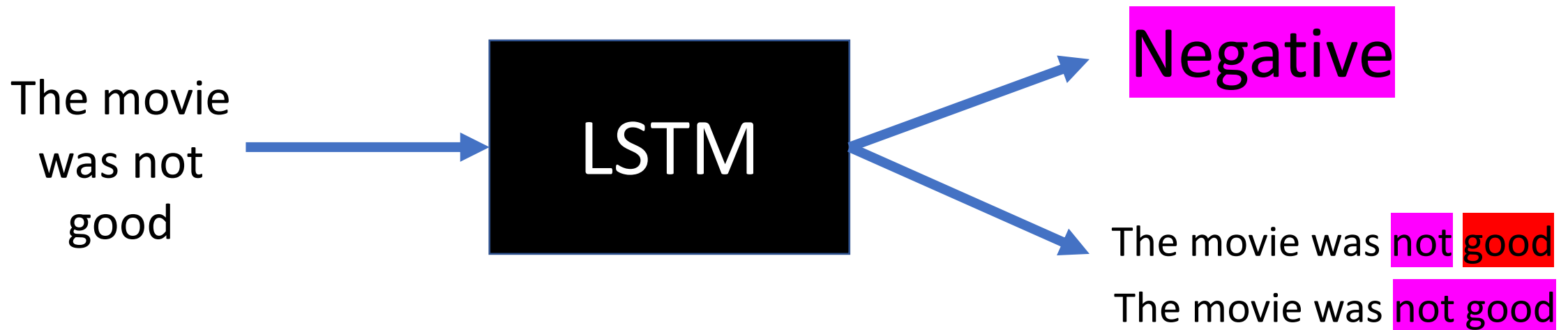
- Underlying relationship involved compositionality



- Predictions requiring compositionality
  - Bag of words fail
- Describing non-linear model → need non-linear explanation

# Solution

- Provide importance scores for every phrase in input
- Phrase scores are not additive
  - Can capture composition
- Different between scores for phrase and constituent words provides interaction



## Contextual Decomposition



- Contextual decomposition for LSTM
- Given subset of input:

$$LSTM(x_1, \dots, x_T) =$$

$$logistic(\beta + \gamma) = \frac{1}{1 + \exp(-(\beta + \gamma))}$$

- $\beta$  corresponds to contributions solely from phrase,  $\gamma$  other factors



# CD of LSTM

- LSTM

$$\begin{aligned}o_t &= \sigma(W_o x_t + V_o h_{t-1} + b_o) \\f_t &= \sigma(W_f x_t + V_f h_{t-1} + b_f) \\i_t &= \sigma(W_i x_t + V_i h_{t-1} + b_i) \\g_t &= \tanh(W_g x_t + V_g h_{t-1} + b_g) \\c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\h_t &= o_t \odot \tanh(c_t)\end{aligned}$$

$$p_j = \text{SoftMax}(W h_T)_j = \frac{\exp(W_j h_T)}{\sum_{k=1}^C \exp(W_k h_T)}$$

$$\begin{aligned}h_t &= \boxed{\beta_t} + \boxed{\gamma_t} \\c_t &= \boxed{\beta_t^c} + \boxed{\gamma_t^c}\end{aligned}$$

- Decomposition for the final output

$$p = \text{SoftMax}(\boxed{W \beta_T} + \boxed{W \gamma_T})$$

Score for the phrase contribution to the  
LSTM prediction

Other factors

# Derivations

- Write each gate of them as a linear sum of contributions from each of their inputs (assume have a way of linearizing)

$$\begin{aligned}i_t &= \sigma(W_i x_t + V_i h_{t-1} + b_i) \\ &= L_\sigma(W_i x_t) + L_\sigma(V_i h_{t-1}) + L_\sigma(b_i)\end{aligned}$$

Try to rewrite c and h as follows:

$$\begin{aligned}c_t &= f_t \odot c_{t-1} + i_t \odot g_t = \beta_t^c + \gamma_t^c \\ h_t &= o_t \odot \tanh(c_t) = \beta_t + \gamma_t\end{aligned}$$

# Disambiguating Interactions Between Gates

$$\begin{aligned}
 f_t \odot c_{t-1} &= (L_\sigma(W_f x_t) + L_\sigma(V_f \beta_{t-1}) + L_\sigma(V_f \gamma_{t-1}) + L_\sigma(b_f)) \odot (\beta_{t-1}^c + \gamma_{t-1}^c) \\
 &= ([L_\sigma(W_f x_t) + L_\sigma(V_f \beta_{t-1}) + L_\sigma(b_f)] \odot \beta_{t-1}^c) \\
 &\quad + (L_\sigma(V_f \gamma_{t-1}) \odot \beta_{t-1}^c + f_t \odot \gamma_{t-1}^c) \\
 &= \beta_t^f + \gamma_t^f
 \end{aligned}$$

$$\begin{aligned}
 i_t \odot g_t &= [L_\sigma(W_i x_t) + L_\sigma(V_i \beta_{t-1}) + L_\sigma(V_i \gamma_{t-1}) + L_\sigma(b_i)] \\
 &\quad \odot [L_{\tanh}(W_g x_t) + L_{\tanh}(V_g \beta_{t-1}) + L_{\tanh}(V_g \gamma_{t-1}) + L_{\tanh}(b_g)] \\
 &= [L_\sigma(W_i x_t) \odot [L_{\tanh}(W_g x_t) + L_{\tanh}(V_g \beta_{t-1}) + L_{\tanh}(b_g)] \\
 &\quad + L_\sigma(V_i \beta_{t-1}) \odot [L_{\tanh}(W_g x_t) + L_{\tanh}(V_g \beta_{t-1}) + L_{\tanh}(b_g)] \\
 &\quad + L_\sigma(b_i) \odot [L_{\tanh}(W_g x_t) + L_{\tanh}(V_g \beta_{t-1})]] \\
 &\quad + [L_\sigma(V_i \gamma_{t-1}) \odot g_t + i_t \odot L_{\tanh}(V_g \gamma_{t-1}) - L_\sigma(V_i \gamma_{t-1}) \odot L_{\tanh}(V_g \gamma_{t-1}) \\
 &\quad + L_\sigma(b_i) \odot L_{\tanh}(b_g)] \\
 &= \beta_t^u + \gamma_t^u
 \end{aligned}$$

# Final Derivations

$$i_t \odot g_t = \beta_t^u + \gamma_t^u$$

$$f_t \odot c_{t-1} = \beta_t^f + \gamma_t^f$$



$$\beta_t^c = \beta_t^f + \beta_t^u$$

$$\gamma_t^c = \gamma_t^f + \gamma_t^u$$

$$h_t = o_t \odot \tanh(c_t)$$

$$= o_t \odot [L_{\tanh}(\beta_t^c) + L_{\tanh}(\gamma_t^c)]$$

$$= \boxed{o_t \odot L_{\tanh}(\beta_t^c)} + \boxed{o_t \odot L_{\tanh}(\gamma_t^c)}$$

$$= \beta_t + \gamma_t$$

# Last: Linearizing Activation Functions

- How to write?

$$\tanh\left(\sum_{i=1}^N y_i\right) = \sum_{i=1}^N L_{\tanh}(y_i)$$

- Telescoping sum

$$L'_{\tanh}(y_k) = \tanh\left(\sum_{j=1}^k y_j\right) - \tanh\left(\sum_{j=1}^{k-1} y_j\right)$$

- Difference of partial sums
- Assume there is natural ordering of  $y_j$


- No clear ordering in LSTM: Average of all possible orderings ( $\pi$ : Permutation)

$$L_{\tanh}(y_k) = \frac{1}{M_N} \sum_{i=1}^{M_N} \left[ \tanh\left(\sum_{j=1}^{\pi_i^{-1}(k)} y_{\pi_i(j)}\right) - \tanh\left(\sum_{j=1}^{\pi_i^{-1}(k)-1} y_{\pi_i(j)}\right) \right]$$

- e.g.:  $N=2$

$$L_{\tanh}(y_1) = \frac{1}{2} ([\tanh(y_1) - \tanh(0)] + [\tanh(y_2 + y_1) - \tanh(y_2)])$$

## Evaluation Setup



- Two standard benchmarks for binary sentiment analysis datasets
- Stanford Sentiment treebank:
  - 10K reviews from rotten tomatoes
  - Human labels for phrase and review level sentiment
- Yelp polarity
  - 600k reviews
  - Labeled as positive/negative if stars are above/below 3

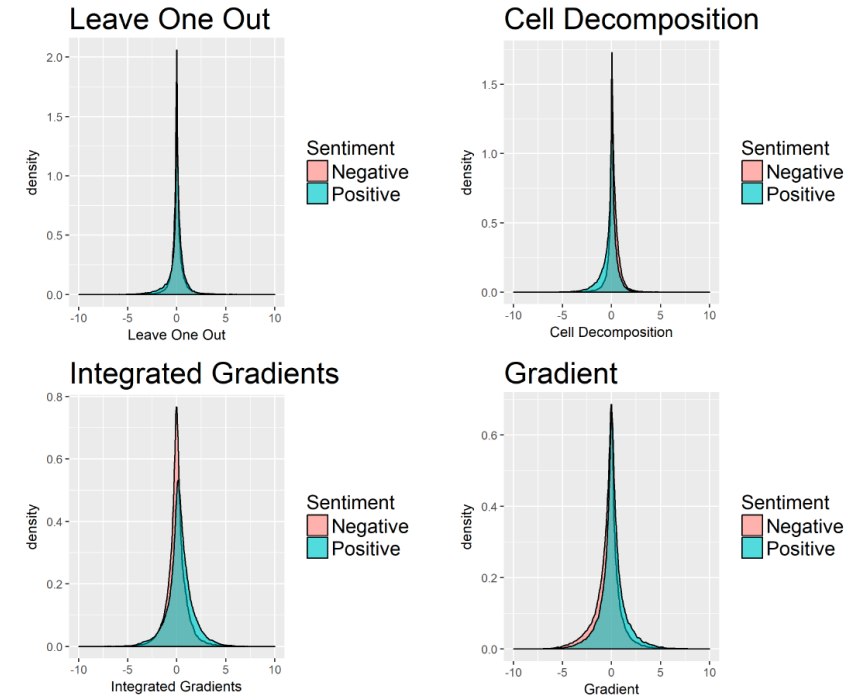
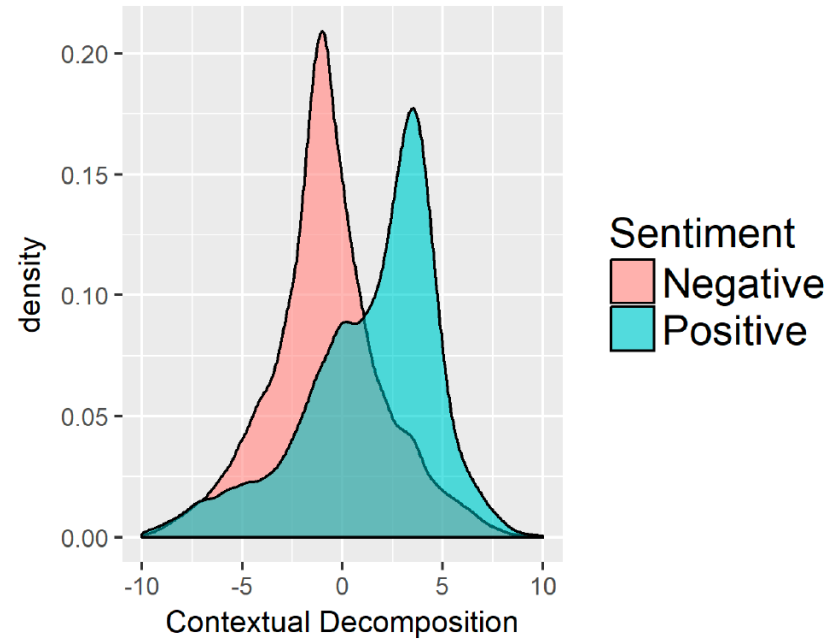
# Short Phrase compositionality

Attribution Method	Heat Map									
Gradient	used	to	be	my	favorite	not	worth	the	time	
Leave One Out (Li et al., 2016)	used	to	be	my	favorite	not	worth	the	time	
Cell decomposition (Murdoch & Szlam, 2017)	used	to	be	my	favorite	not	worth	the	time	
Integrated gradients (Sundararajan et al., 2017)	used	to	be	my	favorite	not	worth	the	time	
Contextual decomposition	used	to	be	my	favorite	not	worth	the	time	

Legend   Very Negative   Negative   Neutral   Positive   Very Positive

- Yelp reviews
  - Positive/negative subphrases contained within phrases of opposing sentiment
- “Favorite” → very positive word, labeled as neutral/negative by prior work

# Contextual Decomposition



## Population-Level Analysis

- Train n-gram model
  - Use it to search for positive/negative subphrases contained within phrases of opposing sentiment
  - The distribution of attributions for positive (negative) sub-phrases contained within negative (positive) phrases



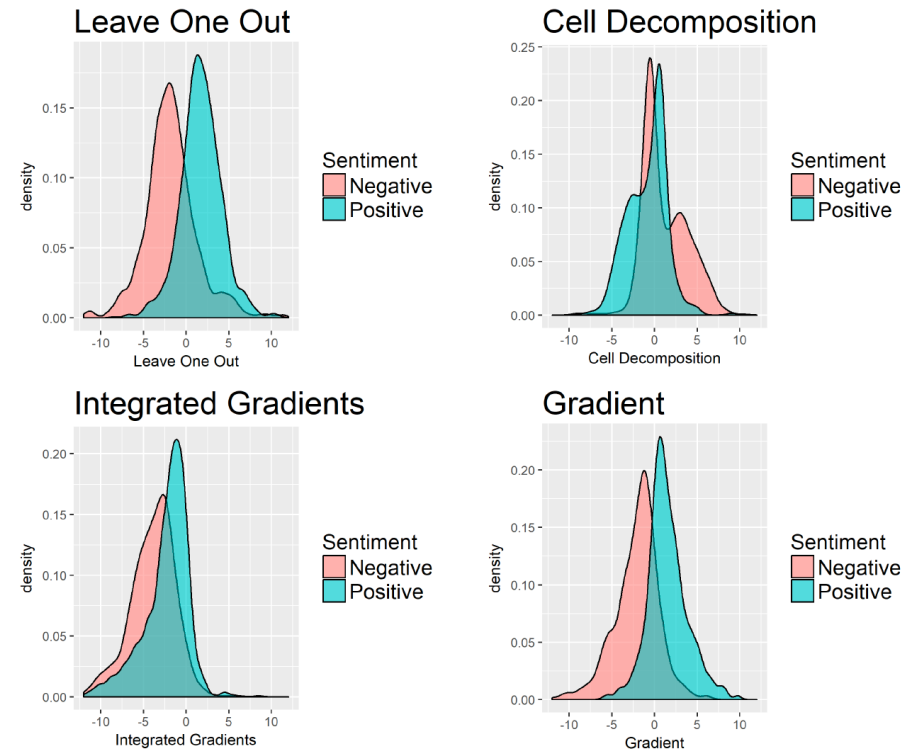
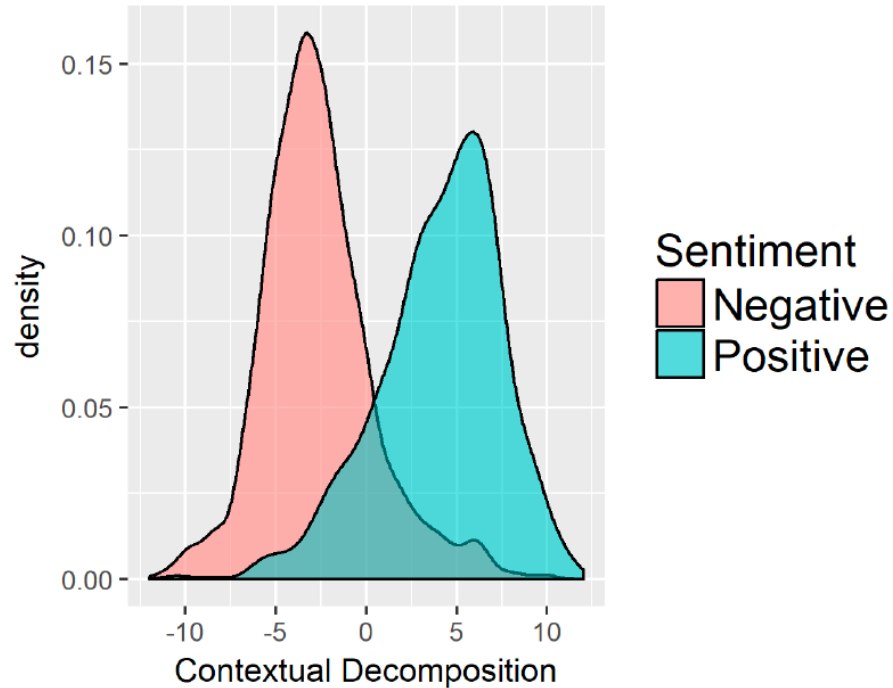
# High-Level Compositionality

Attribution Method	Heat Map
Gradient	It's easy to love Robin Tunney – she's pretty and she can act – but it gets harder and harder to understand her choices.
Leave one out (Li et al., 2016)	It's easy to love Robin Tunney – she's pretty and she can act – but it gets harder and harder to understand her choices.
Cell decomposition (Murdoch & Szlam, 2017)	It's easy to love Robin Tunney – she's pretty and she can act – but it gets harder and harder to understand her choices.
Integrated gradients (Sundararajan et al., 2017)	It's easy to love Robin Tunney – she's pretty and she can act – but it gets harder and harder to understand her choices.
Contextual decomposition	It's easy to love Robin Tunney – she's pretty and she can act – but it gets harder and harder to understand her choices.

Legend Very Negative Negative Neutral Positive Very Positive

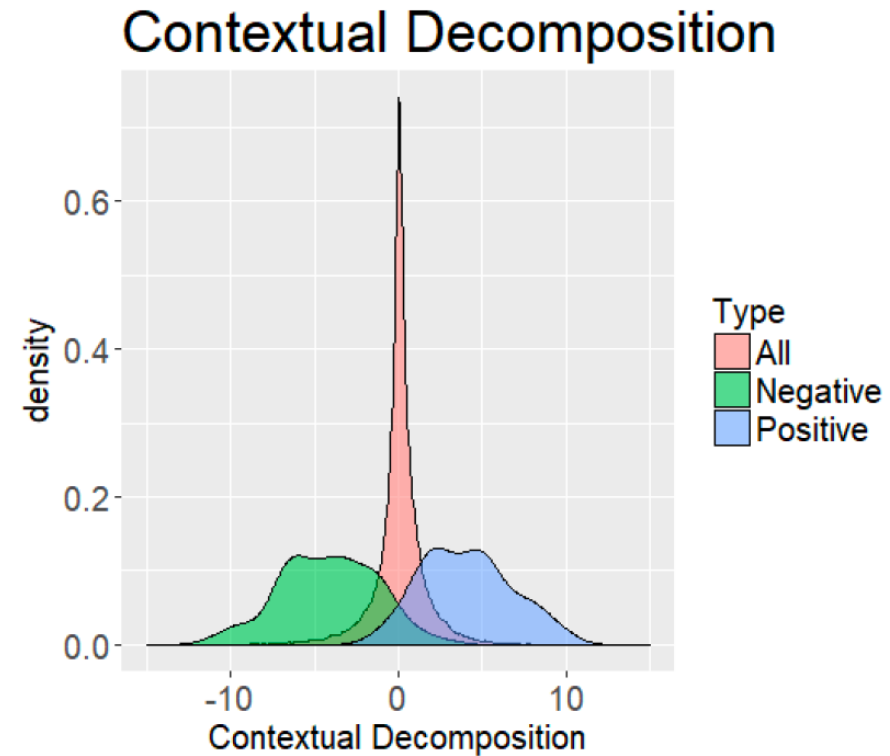
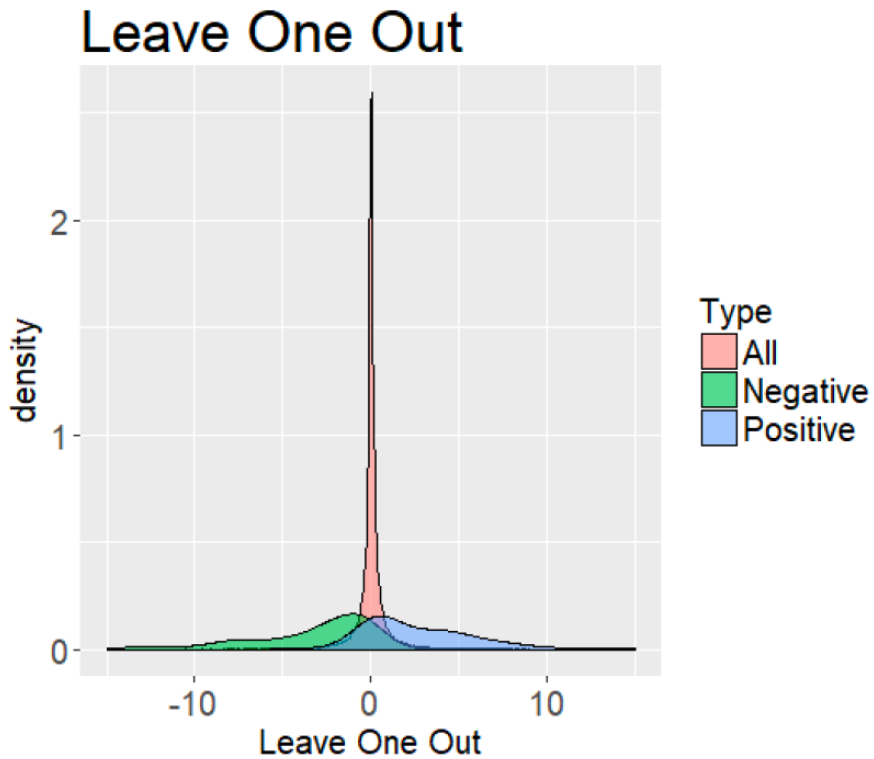
- Stanford sentiment treebank
- Instances where 1/3-2/3 of review has opposite polarity to the review-level label

# Contextual Decomposition



## Population-Level Analysis

- Use phrase-level labels provided by dataset to conduct general search
- Distribution of positive and negative phrases, of length between one and two thirds of the full review



## Extracting negation interactions

- Use Stanford sentiment phrase labels to search for instances of positive/negative negation
  - Negative negation: "This movie was not good"
  - Positive negation: "This move was not bad"
- LSTM learns a negation mechanism

# Conclusion



Introduced contextual decomposition for extracting importance scores for words, phrases and interactions



Demonstrated qualitative improvements of importance scores when dealing with compositionality



Showed, for the first time, LSTMs capture negation