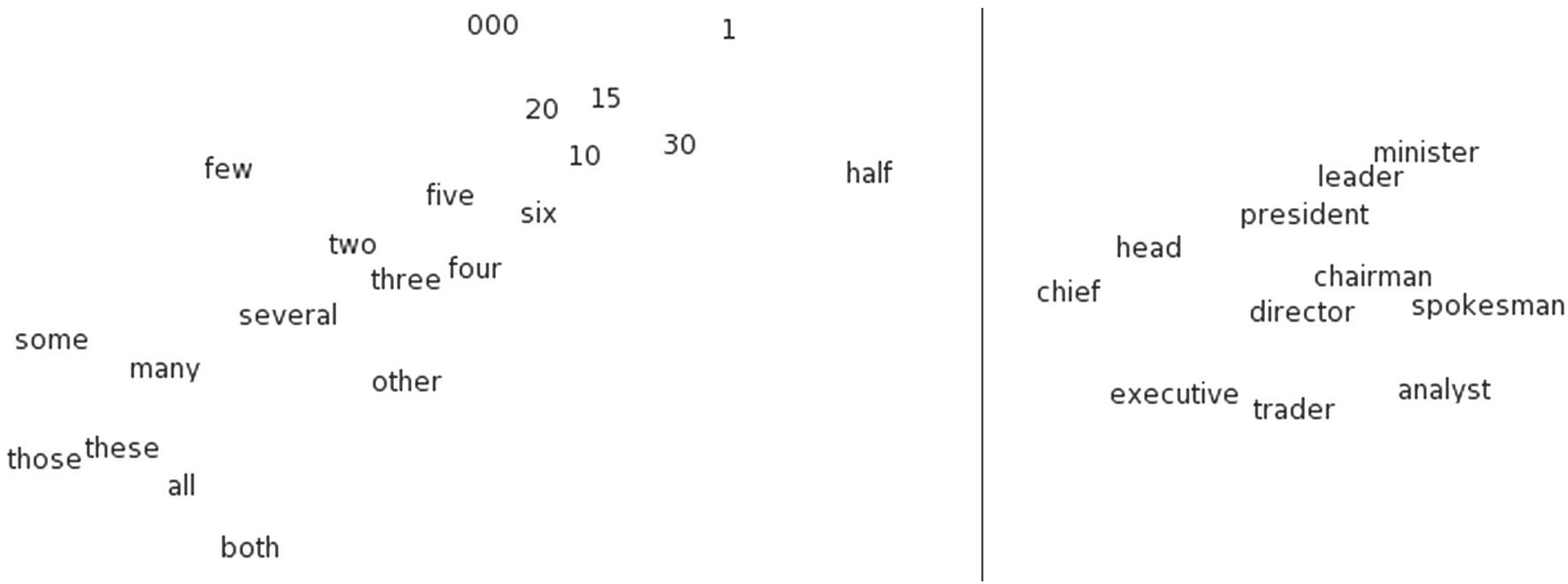
Deep Unordered Composition Rivals Syntactic Methods for Text Classification

Mohit Iyyer, Varun Manjunatha,
Jordan Boyd-Graber, and Hal Daumé III

University of Maryland, College Park
University of Colorado, Boulder

Vector Space Models for NLP

- Represent words by low-dimensional vectors called **embeddings**



From One Word to Many Words

- How do we **compose** word embeddings into vectors that capture the meanings of phrases, sentences, and documents?

From One Word to Many Words

- How do we **compose** word embeddings into vectors that capture the meanings of phrases, sentences, and documents?

From One Word to Many Words

- How do we **compose** word embeddings into vectors that capture the meanings of phrases, sentences, and documents?

Task-Specific Composition Functions

- Sentiment Analysis
- Factoid Question Answering
- Machine Translation
- Parsing
- Image Captioning
- Generation
- Lots more!

Task-Specific Composition Functions

- Sentiment Analysis
- Factoid Question Answering
- Machine Translation
- Parsing
- Image Captioning
- Generation
- Lots more!

Our main contribution: A fast and simple composition function that competes with more complex methods on these two tasks

Outline

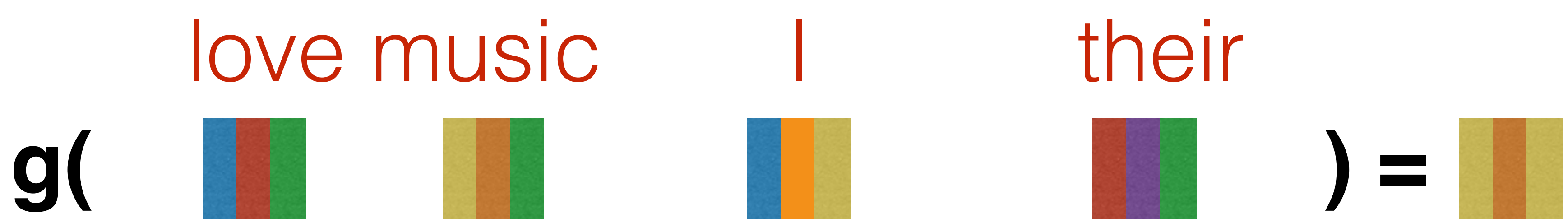
- Review of composition functions
- Deep averaging networks (**DAN**)
- Experiments (factoid QA & sentiment analysis)
- How do **DAN**s work?
- Error analysis & comparisons to previous work

Two Types of Composition

1. Unordered (bag-of-words)

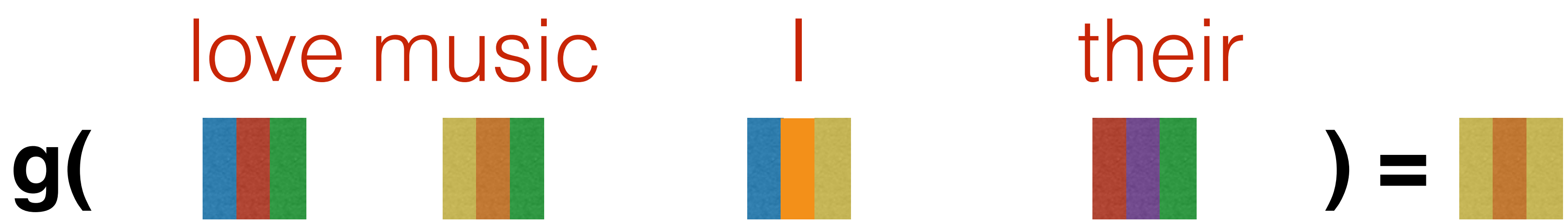
Two Types of Composition

1. Unordered (bag-of-words)



Two Types of Composition

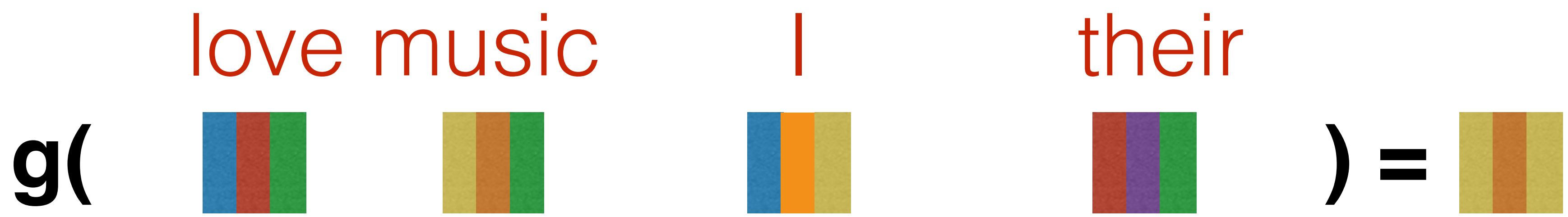
1. Unordered (bag-of-words)



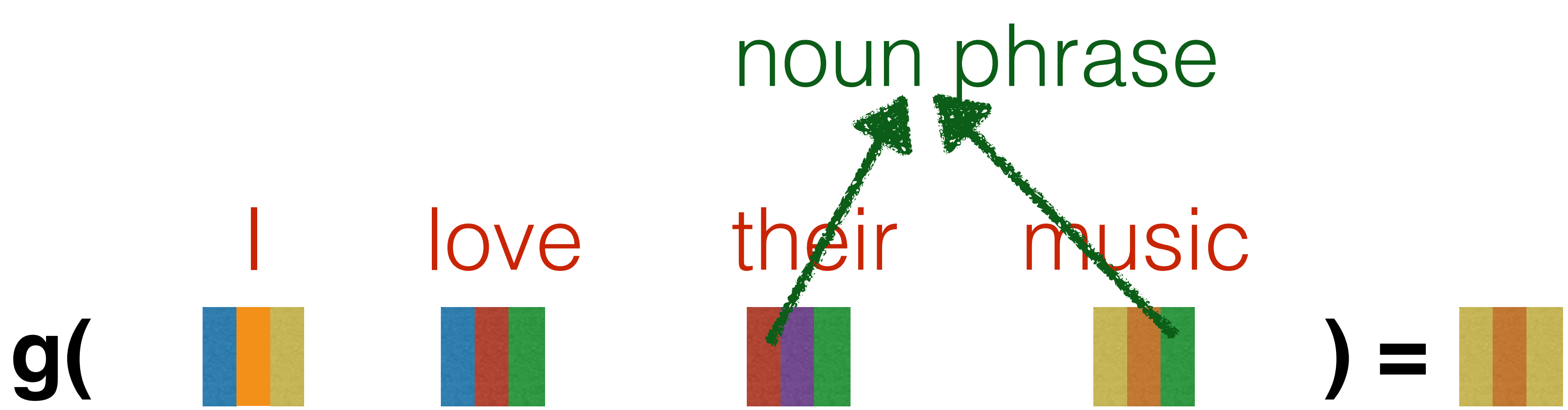
2. Syntactic (incorporates word order and syntax)

Two Types of Composition

1. Unordered (bag-of-words)



2. Syntactic (incorporates word order and syntax)



Unordered Composition: the **NBOW**

- Apply a simple element-wise vector operation to all word embeddings; a **neural bag-of-words**
  - e.g., addition, multiplication, averaging
- Advantages: very fast, simple to implement
- Used previously as a baseline model (e.g., Kalchbrenner & Blunsom, 2014)

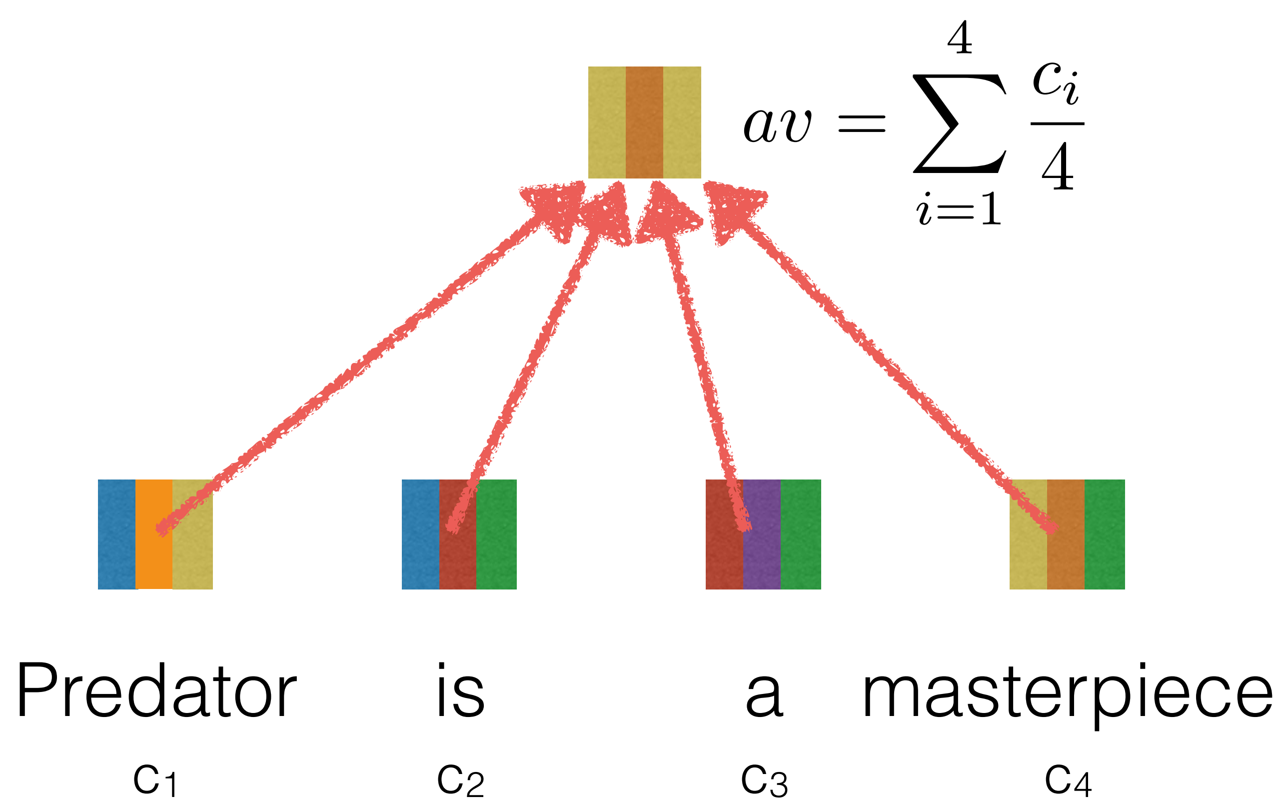
An **NBOW** for Sentiment Analysis

An **NBOW** for Sentiment Analysis

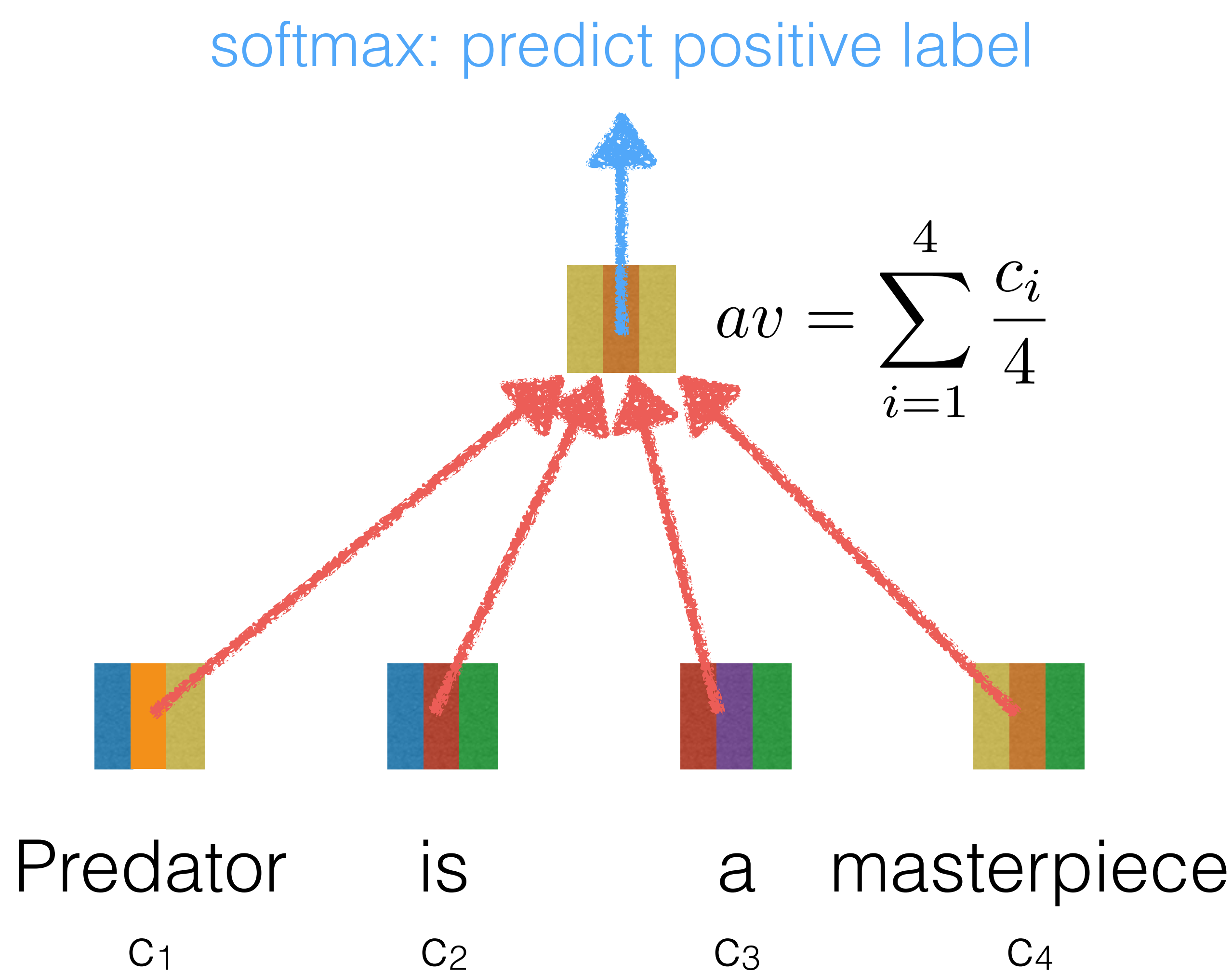
Predator is a masterpiece

c_1 c_2 c_3 c_4

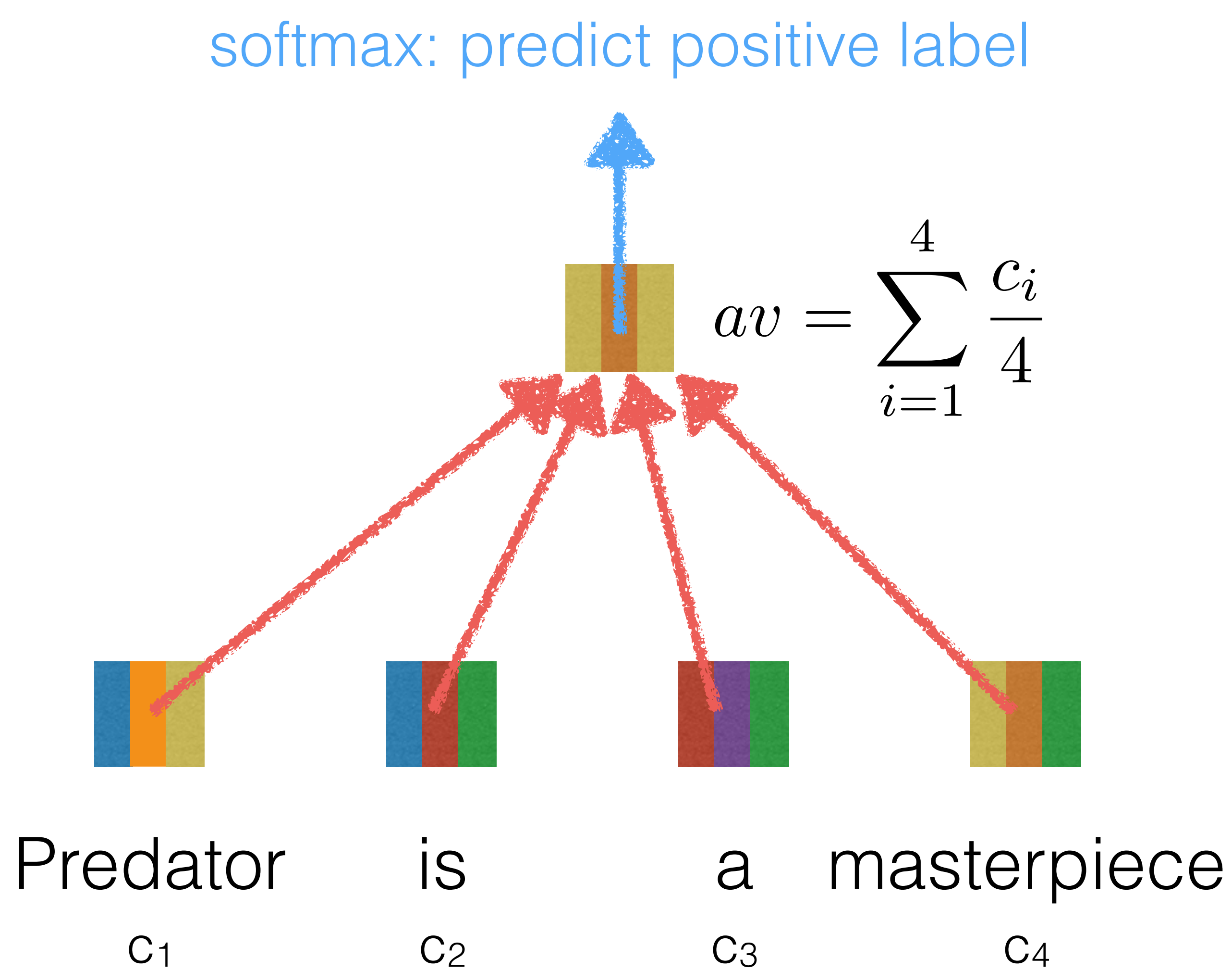
An **NBOW** for Sentiment Analysis



An **NBOW** for Sentiment Analysis



An **NBOW** for Sentiment Analysis



$$av = \sum_{i=1}^{4} \frac{c_i}{4}$$

Relatively low performance on classification tasks!

Syntactic Composition

- Neural network-based approaches
  - Recursive
  - Recurrent
  - Convolutional
- Advantages: usually yield higher accuracies than unordered functions on downstream tasks

Syntactic Composition

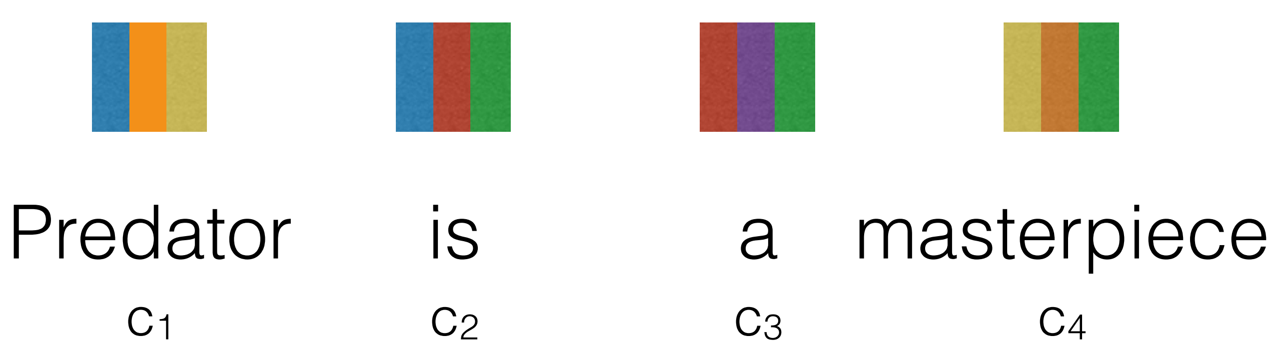
- Neural network-based approaches
    - Recursive
    - Recurrent
    - Convolutional
- Advantages: usually yield higher accuracies than unordered functions on downstream tasks

Recursive Neural Networks (**RecNN**)

- **g** depends on a *parse tree* of the input text sequence

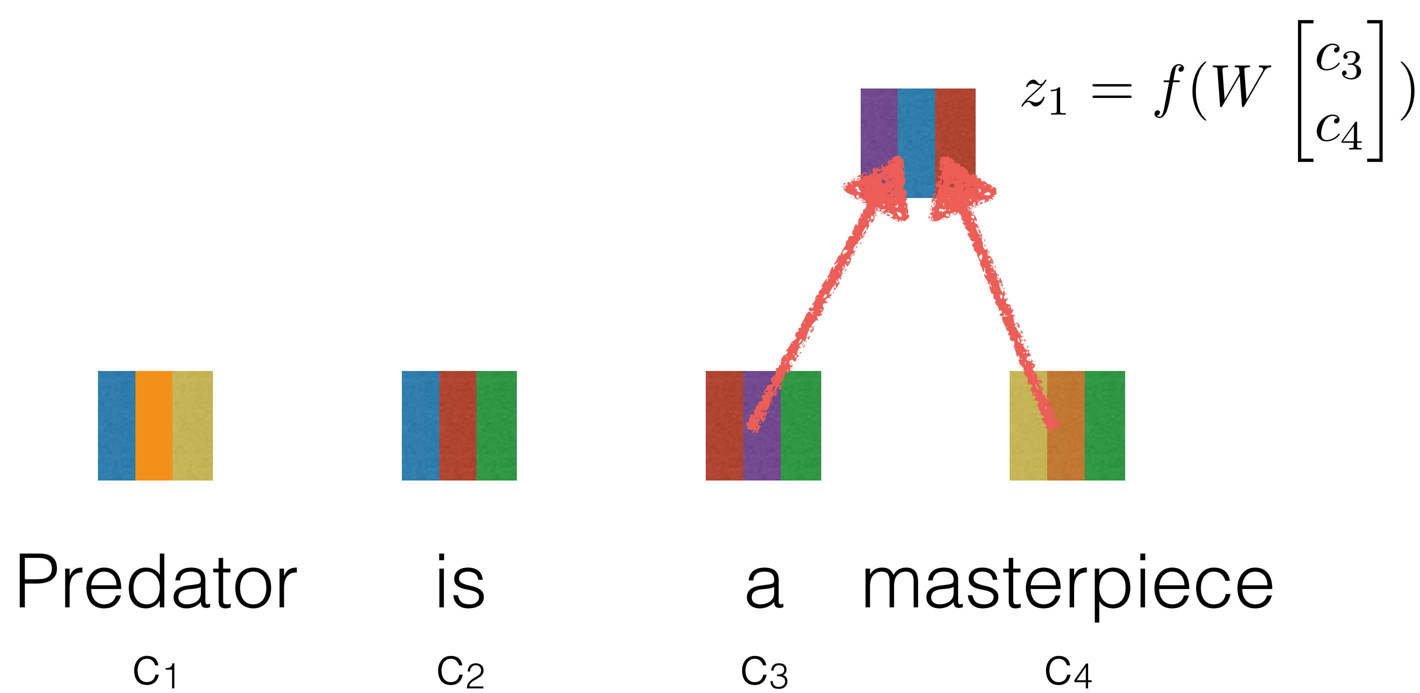
Recursive Neural Networks (**RecNN**)

- **g** depends on a *parse tree* of the input text sequence



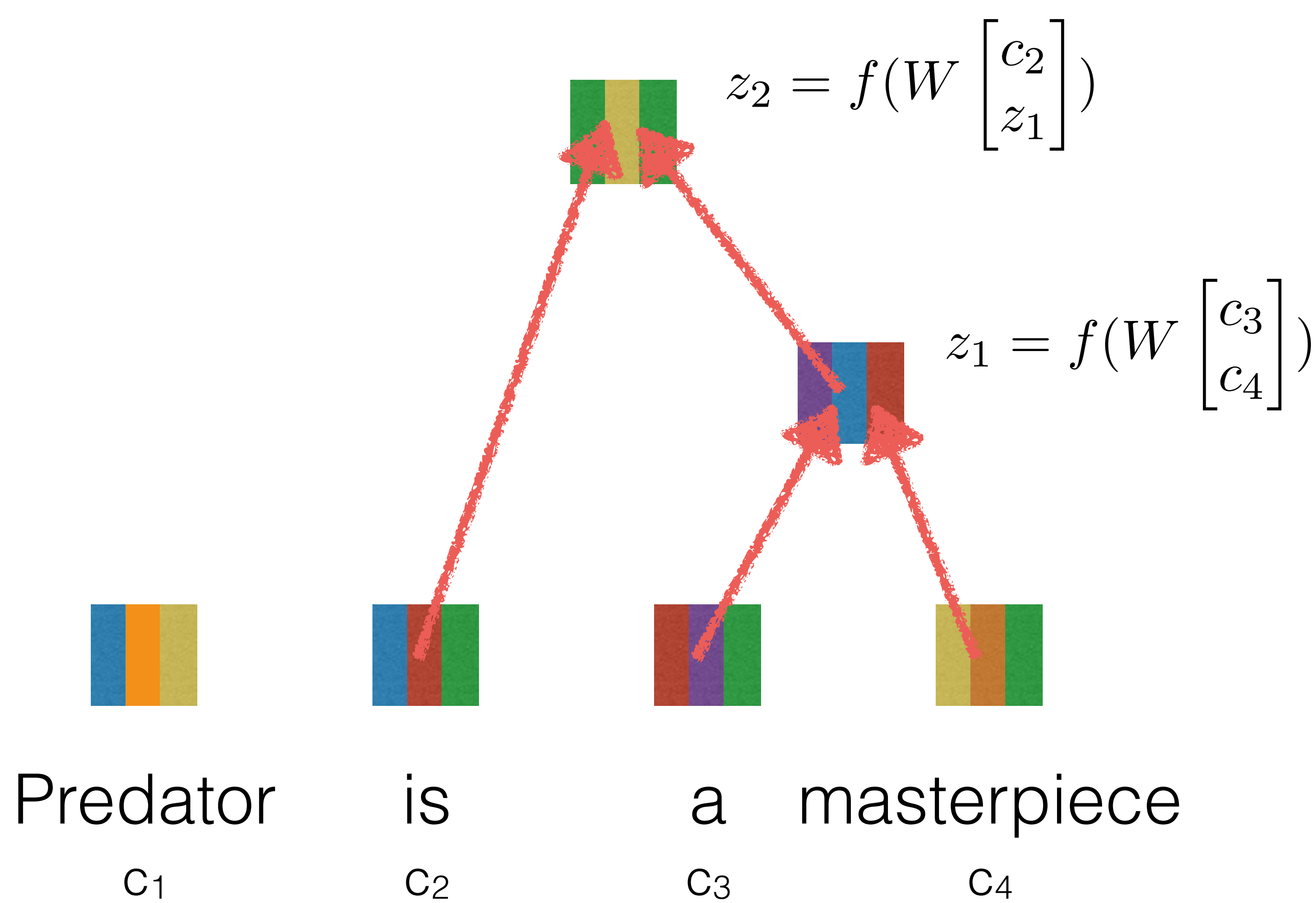
Recursive Neural Networks (**RecNN**)

- **g** depends on a *parse tree* of the input text sequence



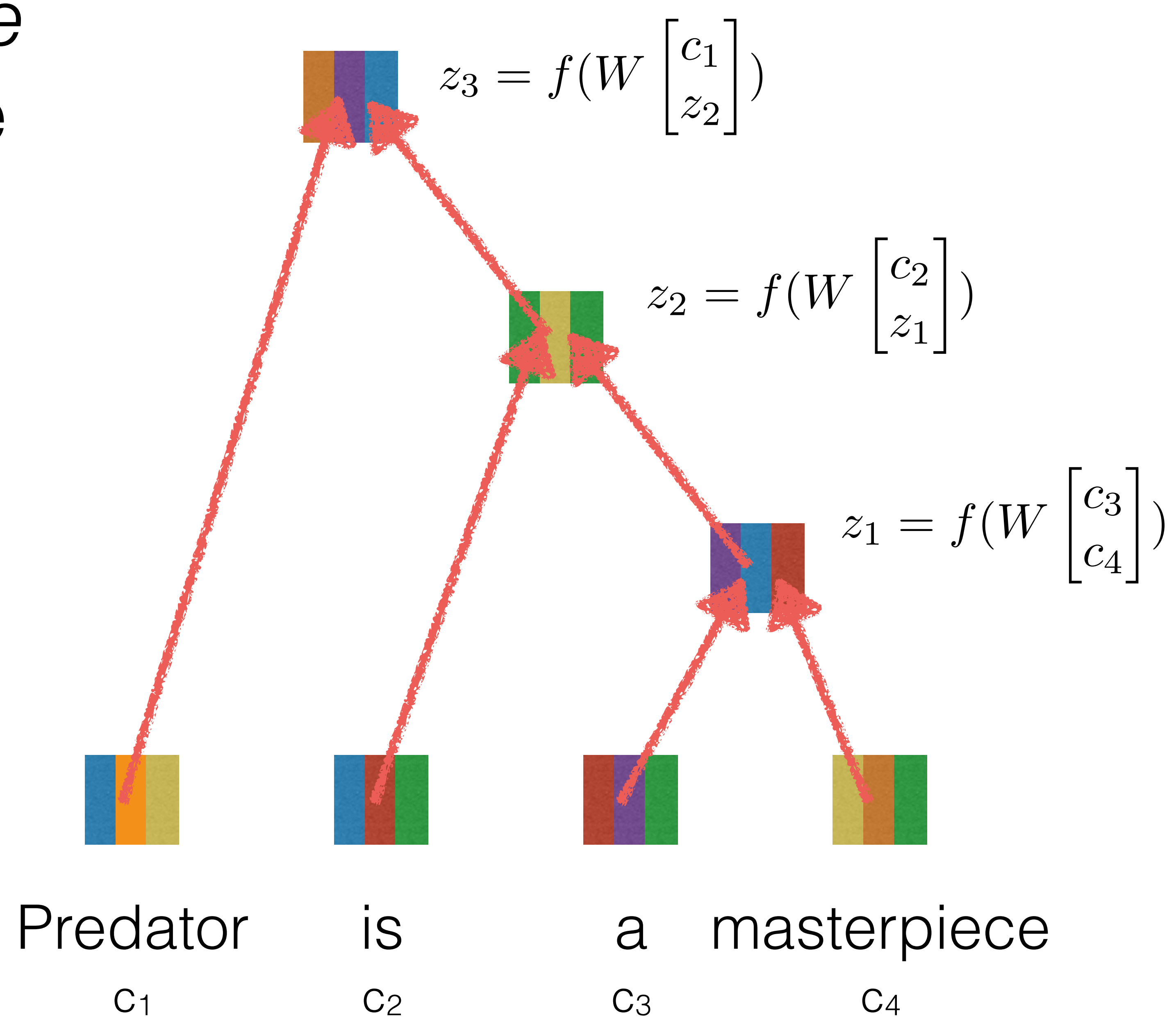
Recursive Neural Networks (**RecNN**)

- **g** depends on a *parse tree* of the input text sequence



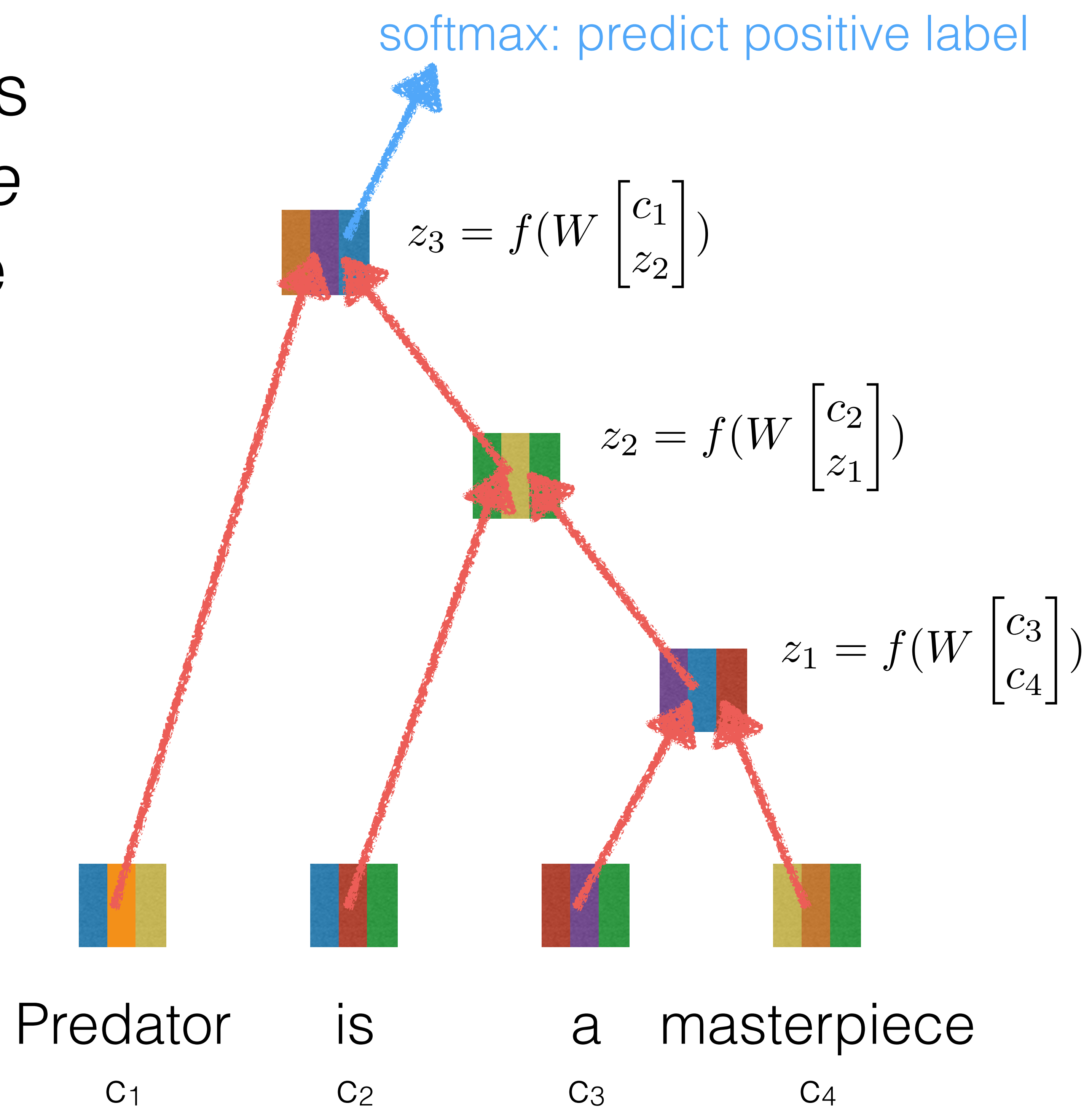
Recursive Neural Networks (**RecNN**)

- **g** depends on a *parse tree* of the input text sequence



Recursive Neural Networks (**RecNN**)

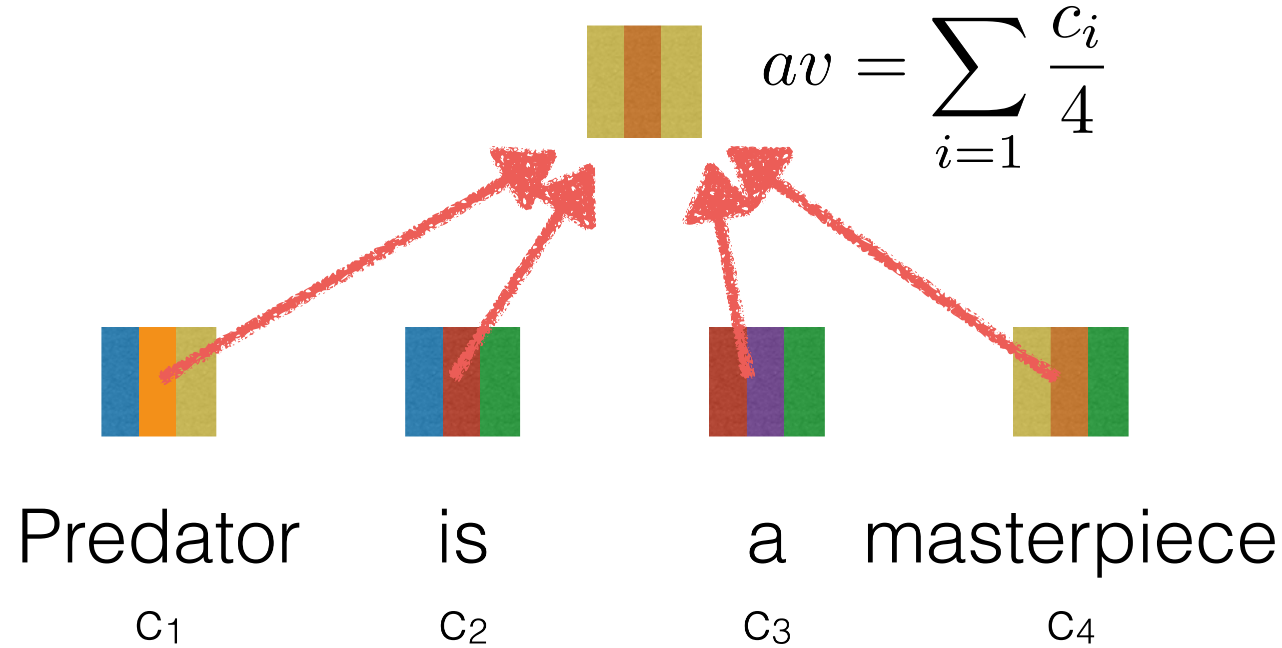
- **g** depends on a *parse tree* of the input text sequence



Isolating the Impact of Syntax

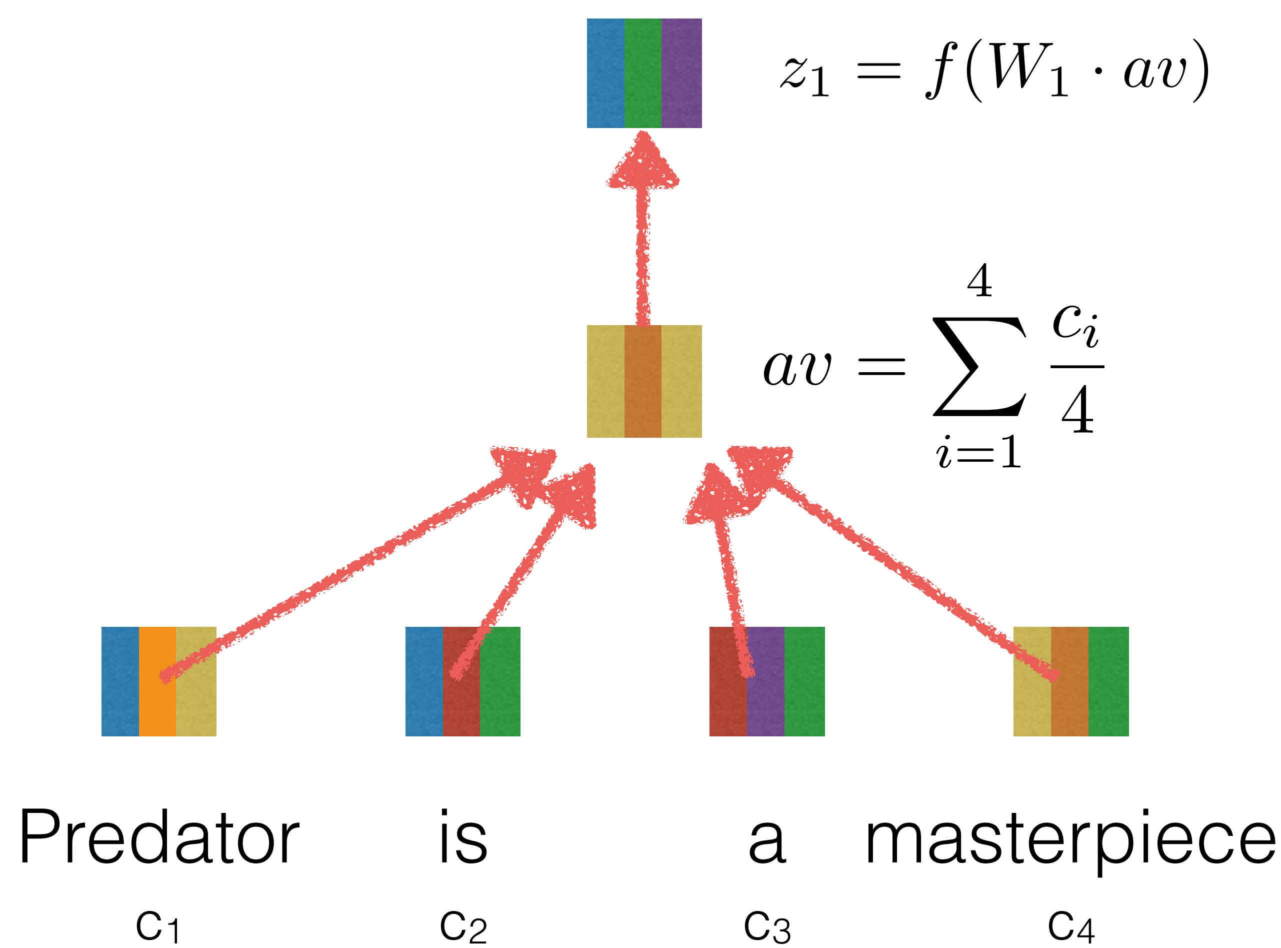
- **RecNN**s have two advantages over **NBOW** models: syntax (obviously) and *nonlinear transformations*
- removing nonlinearities from **RecNN**s decreases absolute sentiment classification accuracy by over 5% (Socher et al., 2013)
- **NBOW**s are linear mappings between embeddings and outputs… what happens if we add nonlinearities?

Deep Averaging Networks

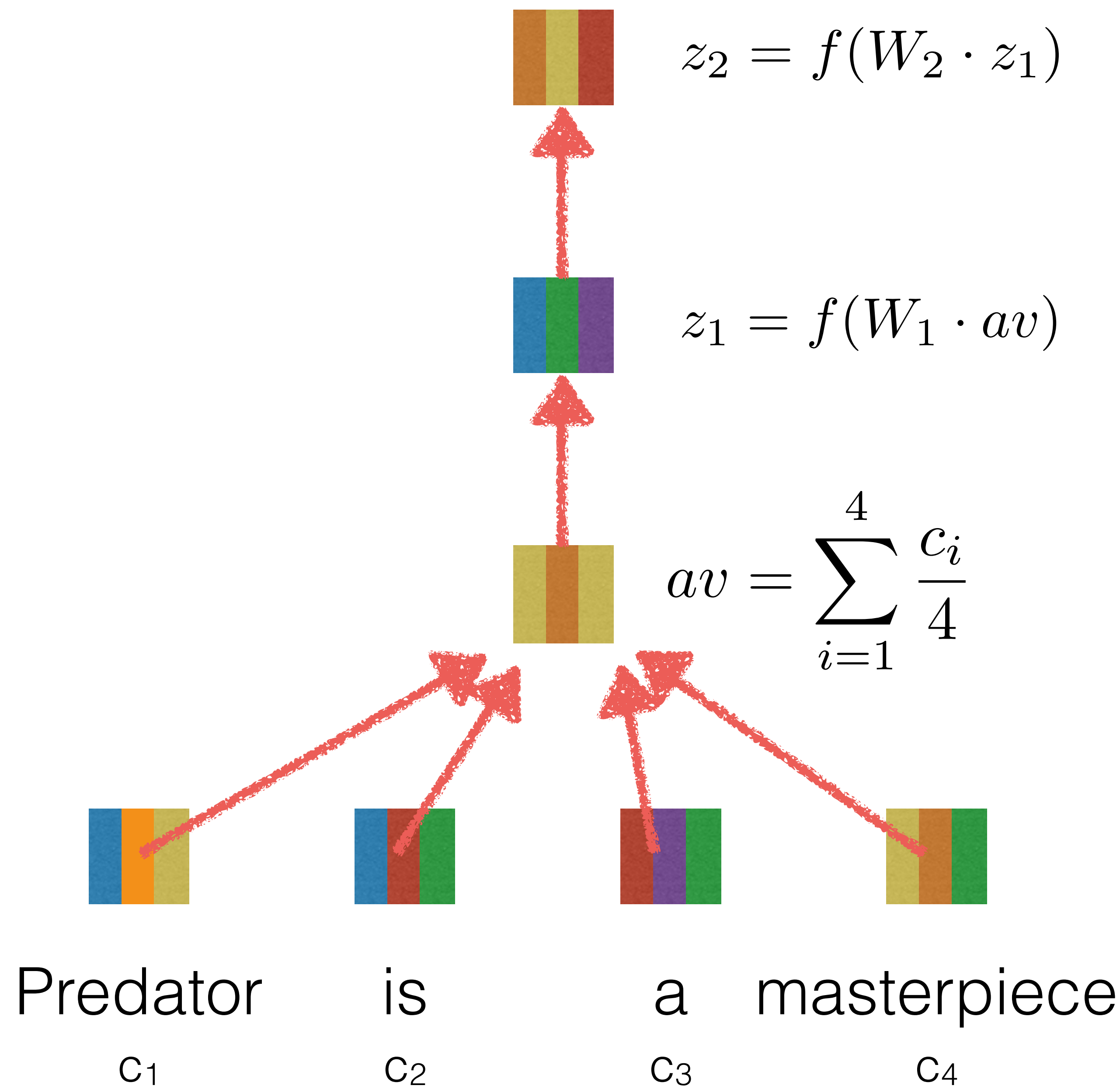


$$av = \sum_{i=1}^{4} \frac{c_i}{4}$$

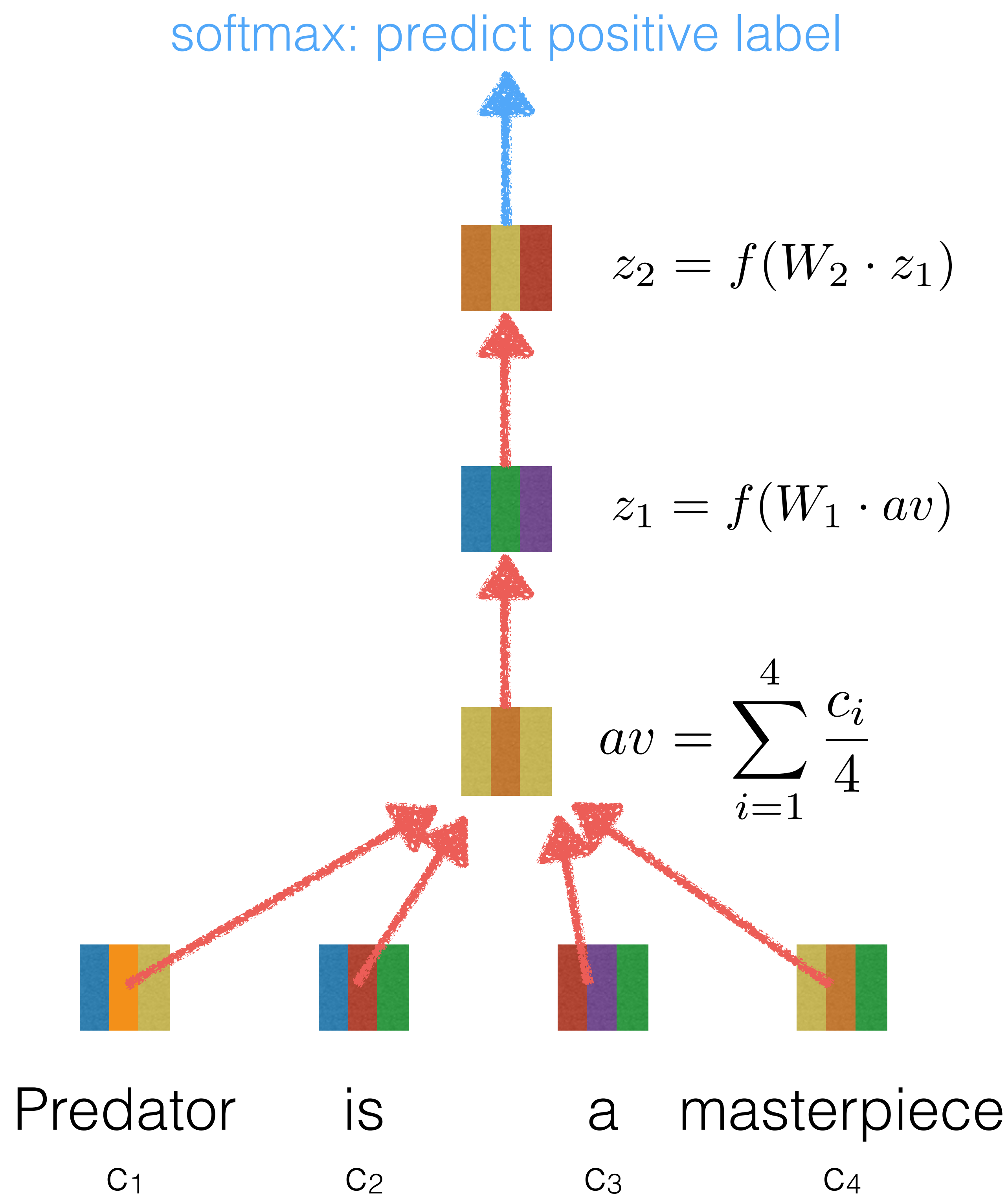
Deep Averaging Networks



Deep Averaging Networks



Deep Averaging Networks



Experiments

Factoid Question Answering

Sentiment Analysis

QA: Quiz Bowl

QA: Quiz Bowl

This creature has female counterparts named Penny and Gown.

QA: Quiz Bowl

This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.

QA: Quiz Bowl

This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.

This creature's counterparts include Daemon on the Berkeley Software Distribution, or BSD.

QA: Quiz Bowl

This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.

This creature's counterparts include Daemon on the Berkeley Software Distribution, or BSD.

For ten points, name this mascot of the Linux operating system, a penguin whose name refers to formal male attire.

QA: Quiz Bowl

This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.

This creature's counterparts include Daemon on the Berkeley Software Distribution, or BSD.

For ten points, name this mascot of the Linux operating system, a penguin whose name refers to formal male attire.

Answer: Tux

QA: Dataset

- Used in this work: history quiz bowl question dataset of Iyyer et al., 2014
  - *original dataset*: 3,761 question/answer pairs
  - +*wiki* dataset: original + 53,234 sentence/page-title pairs from Wikipedia

QA: Models

- **BoW-DT:** bag-of-unigrams logistic regression with dependency relations
- **IR**: an information retrieval system built with Whoosh, uses BM-25 term weighting, query expansion, and fuzzy query matching
- **QANTA**: a recursive neural network structured around dependency parse trees
- **DAN**: our model with three hidden layers, trained with *word dropout* regularization

QA: Results

Model	Pos 1	Pos 2	Full	Time (sec)
BoW-DT	35.4	57.7	60.2	—
IR	37.5	65.9	71.4	N/A
QANTA	47.1	72.1	73.7	314
DAN	46.4	70.8	71.8	**18**

QA: Results

Model	Pos 1	Pos 2	Full	Time (sec)
BoW-DT	35.4	57.7	60.2	—
IR	37.5	65.9	71.4	N/A
QANTA	47.1	72.1	73.7	314
DAN	46.4	70.8	71.8	**18**
IR-WIKI	53.7	**76.6**	**77.5**	N/A
QANTA-WIKI	46.5	72.8	73.9	1,648
DAN-WIKI	**54.8**	75.5	77.1	119

QA: Results

Model	Pos 1	Pos 2	Full	Time (sec)
BoW-DT	35.4	57.7	60.2	—
IR	37.5	65.9	71.4	N/A
QANTA	47.1	72.1	73.7	314
DAN	46.4	70.8	71.8	**18**
IR-WIKI	53.7	**76.6**	**77.5**	N/A
QANTA-WIKI	46.5	72.8	73.9	1,648
DAN-WIKI	**54.8**	75.5	77.1	119

DANs Handle Syntactic Diversity

- Sentences from Wikipedia are syntactically different from quiz bowl questions

QB: "Identify this British author who wrote Wuthering Heights" ⟶ very common imperative construction in QB

- They can also contain lots of noise!

WIKI: "She does not seem to have made any friends outside her family." (from *Emily Brontë*'s page)

QA: Man vs. Machine

- Scaled up a **DAN** (in combination with language model features) to handle ~100k Q/A pairs with ~14k unique answers!
- Our system played a match against a team of four former multiple-day Jeopardy champions

QA: Man vs. Machine

- Scaled up a **DAN** (in combination with language model features) to handle ~100k Q/A pairs with ~14k unique answers!
- Our system played a match against a team of four former multiple-day Jeopardy champions

The result: a 200-200 **tie**!

QA: Man vs. Machine

- Scaled up a **DAN** (in combination with language model features) to handle ~100k Q/A pairs with ~14k unique answers!
- Our system played a match against a team of four former multiple-day Jeopardy champions

The result: a 200-200 **tie**!

Round 2 in October: our system duels Ken Jennings

Silly humans…

Sentiment: Datasets

- Sentence-level:
  - Rotten Tomatoes (**RT**) movie reviews (Pang & Lee, 2005): 5,331 positive and 5,331 negative sentences
  - Stanford Sentiment Treebank (**SST**) (Socher et al., 2013): modified version of **RT** with fine-grained phrase annotations
- Document-level:
  - IMDB movie review dataset (Maas et al., 2011): 12,500 positive reviews and 12,500 negative reviews

Sentiment: Syntactic Models

- Standard **RecNNs** and more powerful variants: **deep RecNN** (Irsoy & Cardie, 2014), **RecNTN** (Socher et al., 2013)
- Standard convolutional nets (**CNN-MC** of Kim, 2014) and **dynamic CNN**s (Kalchbrenner et al., 2014)
- Paragraph vector (Le & Mikolov, 2014), restricted Boltzmann machine (Dahl et al., 2012)

Sentiment: Results

Model	RT	SST fine	SST binary	IMDB	Time (sec)
DAN	80.3	47.7	86.3	89.4	136
NBOW	79.0	43.6	83.6	89.0	**91**

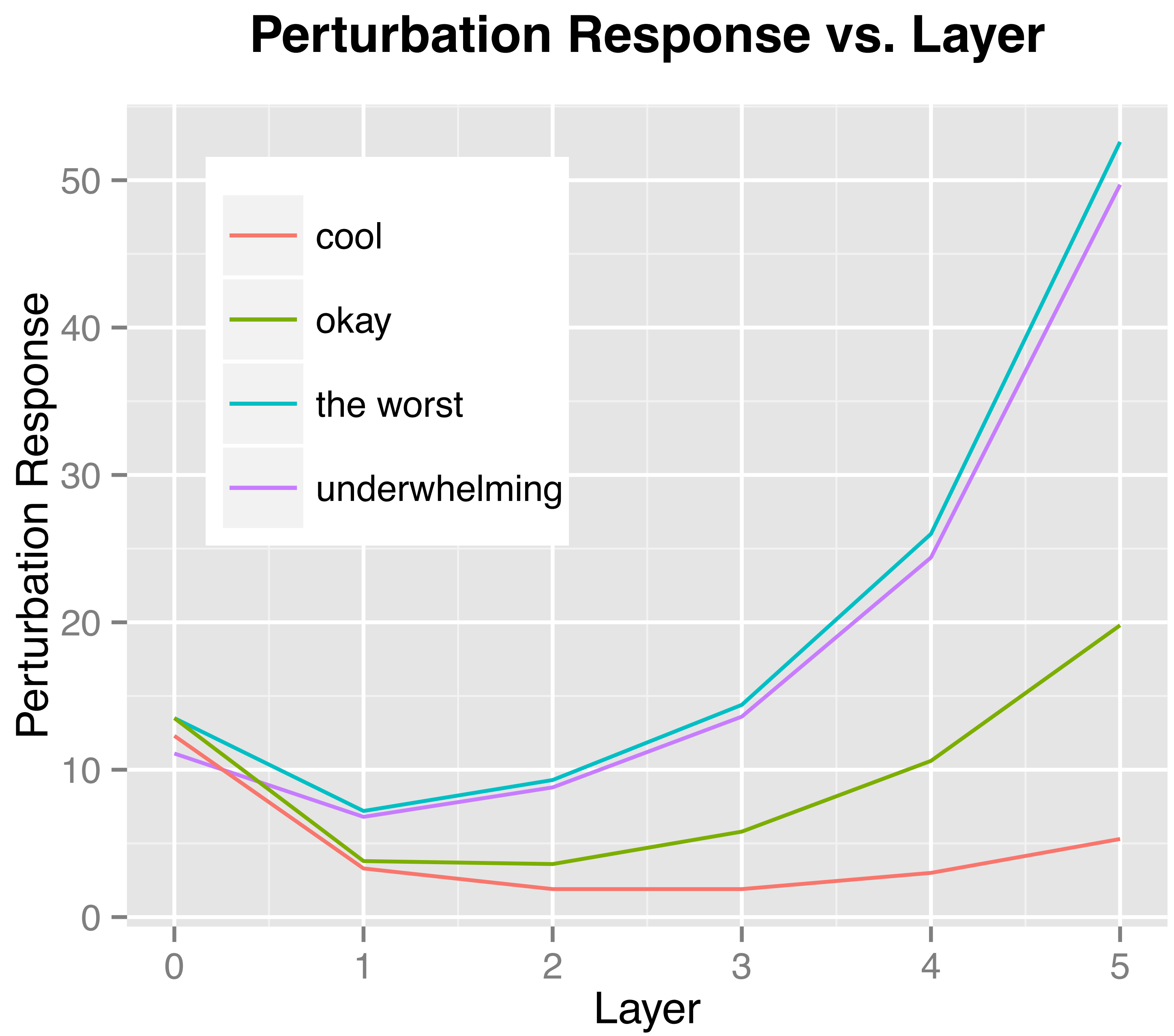
Sentiment: Results

Model	RT	SST fine	SST binary	IMDB	Time (sec)
DAN	80.3	47.7	86.3	89.4	136
NBOW	79.0	43.6	83.6	89.0	**91**
RecNN	77.7	43.2	82.4	—	—
RecNTN	—	45.7	85.4	—	—
DRecNN	—	49.8	86.6	—	431
TreeLSTM	—	**50.6**	86.9	—	—
DCNN	—	48.5	86.9	89.4	—
PVEC	—	48.7	87.8	**92.6**	—
CNN-MC	**81.1**	47.4	**88.1**	—	2,452
WRRBM	—	—	—	89.2	—

How do **DAN**s work?

How do **DAN**s work?

- The film's performances were awesome



What About Negations?

- We collect 48 positive and 44 negative sentences from the SST that each contain at least one negation and one contrastive conjunction
- When confronted with a negation, both the unordered **DAN** and syntactic **DRecNN** predict negative sentiment around 70% of the time.
- Accuracy on only the positive sentences in our subset is low: 37.5% for the **DAN** and 41.7% for the **DRecNN**

Sentence	DAN	DRecNN	Ground-Truth
blessed with immense physical prowess he may well be, but ahola is simply not an actor	positive	neutral	negative
too bad , but thanks to some lovely comedic moments and several fine performances, it's not a total loss	negative	negative	positive
it's so good that its relentless, polished wit can withstand not only inept school productions, but even oliver parker's movie adaptation	negative	positive	positive
the movie was bad	negative	negative	negative
the movie was not bad	negative	negative	positive

Recap

- Introduced the **DAN** for fast and simple text classification

- Our findings suggest that nonlinearly transforming input embeddings is crucial for performance

- Complex syntactic models make mistakes similar to those of the more naïve **DAN**s… syntax is important, but we need more data and/or models that generalize with fewer examples

Thanks! Questions?

code@github.com/miyyer/dan