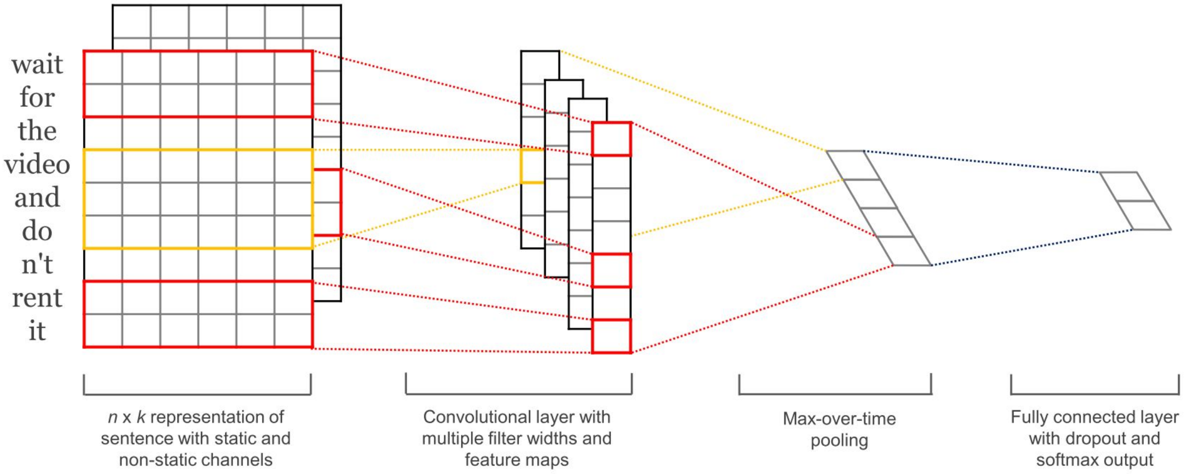
Understanding Convolutional Neural Networks for Text Classification

Alon Jacovi, Oren Sar Shalom, Yoav Goldberg

Introduction

CNNs are often used for **Text Classification**.

Very effective, even with a single layer (*Kim, 2014*).



Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification

Introduction

How do CNN classifiers process text?

What functions do they learn?

What abstractions or reasoning do they make on the data?

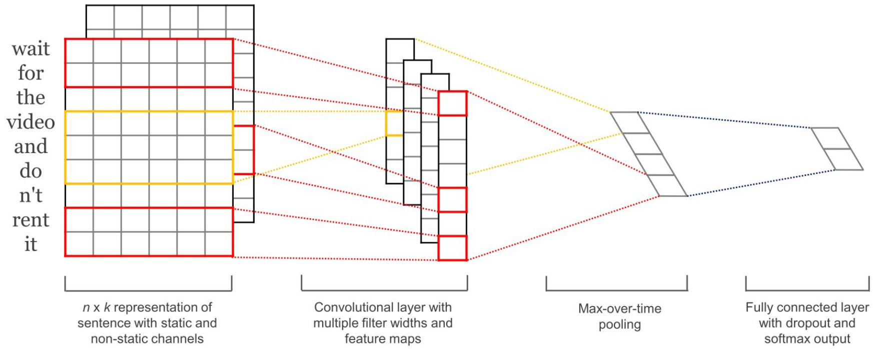
This Work

We aim to understand the dynamics of CNNs for text classification.

Two main questions:

1. Which ngrams contribute to classification? (*prediction interpretability*)
2. What does each filter capture? (*model interpretability*)

All of the examples are from sentiment classification (three datasets).



Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification

Background
Single-layer 1D CNNs

wait for the video and do n't rent it

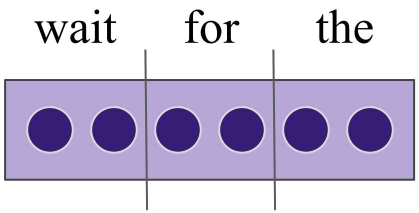
(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$

$$i \le n - \ell$$

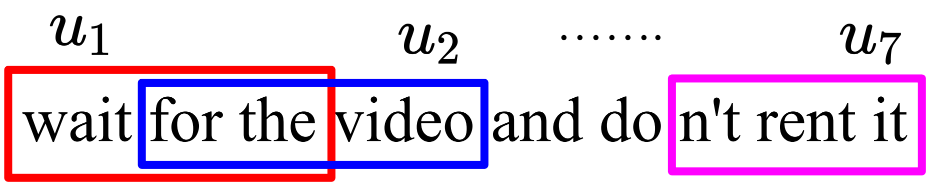
u_1

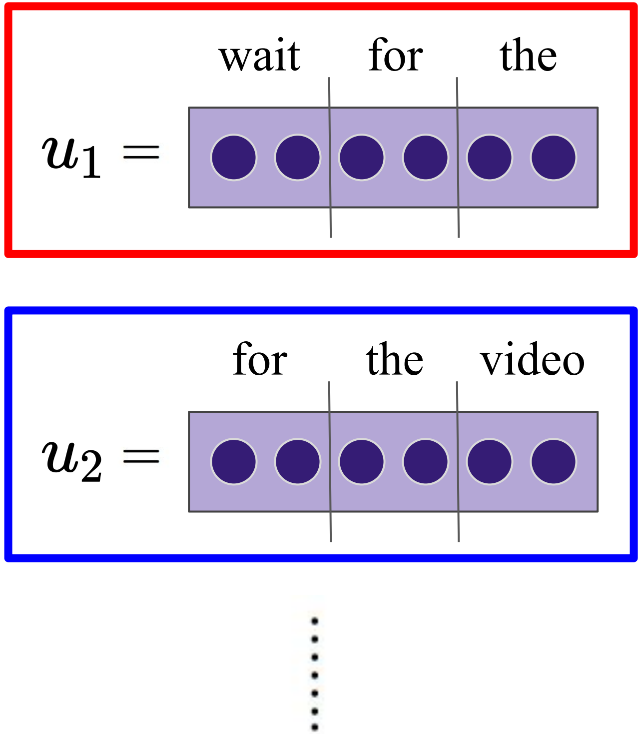
$\boxed{\text{wait for the}}$ video and do n't rent it

$u_1 =$ 

(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \le n - \ell$$





(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \leq n - \ell$$

u_1 u_2 ······· u_7

wait for the video and do n't rent it

$\langle \mathbf{u}_1, \mathbf{f}_1 \rangle$ **0.7**

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$

$$i \le n - \ell$$

u_1 u_2 .... u_7

wait for the video and don't rent it

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$\langle \mathbf{u}_1, \mathbf{f}_1 \rangle$ **0.7**

$\langle \mathbf{u}_1, \mathbf{f}_2 \rangle$ **3.2**

$\langle \mathbf{u}_1, \mathbf{f}_3 \rangle$ **5.1**

(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \le n - \ell$$

u_1 u_2 .... u_7

wait for the video and do n't rent it

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$\langle \mathbf{u}_1, \mathbf{f}_1 \rangle$	0.7	$\langle \mathbf{u}_2, \mathbf{f}_1 \rangle$	1.3
$\langle \mathbf{u}_1, \mathbf{f}_2 \rangle$	3.2	$\langle \mathbf{u}_2, \mathbf{f}_2 \rangle$	5.2
$\langle \mathbf{u}_1, \mathbf{f}_3 \rangle$	5.1	$\langle \mathbf{u}_2, \mathbf{f}_3 \rangle$	-0.9

(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \leq n - \ell$$

u_1 u_2 ……. u_7

wait for the video and do n't rent it

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$\langle \mathbf{u}_1, \mathbf{f}_1 \rangle$	0.7	$\langle \mathbf{u}_2, \mathbf{f}_1 \rangle$	1.3		$\langle \mathbf{u}_7, \mathbf{f}_1 \rangle$	1.1
$\langle \mathbf{u}_1, \mathbf{f}_2 \rangle$	3.2	$\langle \mathbf{u}_2, \mathbf{f}_2 \rangle$	5.2	…….	$\langle \mathbf{u}_7, \mathbf{f}_2 \rangle$	8.5
$\langle \mathbf{u}_1, \mathbf{f}_3 \rangle$	5.1	$\langle \mathbf{u}_2, \mathbf{f}_3 \rangle$	-0.9		$\langle \mathbf{u}_7, \mathbf{f}_3 \rangle$	4.6

(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \le n - \ell$$

u_1 u_2 u_7

wait for the video and do n't rent it

$\langle \mathbf{u}_1, \mathbf{f}_1 \rangle$ 0.7	$\langle \mathbf{u}_2, \mathbf{f}_1 \rangle$ 1.3		$\langle \mathbf{u}_7, \mathbf{f}_1 \rangle$ 1.1	**max**
$\langle \mathbf{u}_1, \mathbf{f}_2 \rangle$ 3.2	$\langle \mathbf{u}_2, \mathbf{f}_2 \rangle$ 5.2	....	$\langle \mathbf{u}_7, \mathbf{f}_2 \rangle$ 8.5	
$\langle \mathbf{u}_1, \mathbf{f}_3 \rangle$ 5.1	$\langle \mathbf{u}_2, \mathbf{f}_3 \rangle$ -0.9		$\langle \mathbf{u}_7, \mathbf{f}_3 \rangle$ 4.6	

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$$p_j = \mathrm{ReLU}(\max_i F_{ij})$$

← max pooling

(*word embeddings*)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \leq n - \ell$$

u_1 u_2 ……. u_7

wait for the video and don't rent it

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$\langle \mathbf{u}_1, \mathbf{f}_1 \rangle$	0.7	$\langle \mathbf{u}_2, \mathbf{f}_1 \rangle$	1.3		$\langle \mathbf{u}_7, \mathbf{f}_1 \rangle$	1.1
$\langle \mathbf{u}_1, \mathbf{f}_2 \rangle$	3.2	$\langle \mathbf{u}_2, \mathbf{f}_2 \rangle$	5.2	…….	$\langle \mathbf{u}_7, \mathbf{f}_2 \rangle$	8.5
$\langle \mathbf{u}_1, \mathbf{f}_3 \rangle$	5.1	$\langle \mathbf{u}_2, \mathbf{f}_3 \rangle$	-0.9		$\langle \mathbf{u}_7, \mathbf{f}_3 \rangle$	4.6

max

$$p_j = \mathrm{ReLU}(\max_i F_{ij})$$

for the video
$\langle u_2, f_1 \rangle$
1.3

$\leftarrow$ max pooling

(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \leq n - \ell$$

u_1 u_2 u_7

wait for the video and do n't rent it

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$\langle \mathbf{u}_1, \mathbf{f}_1 \rangle$ 0.7	$\langle \mathbf{u}_2, \mathbf{f}_1 \rangle$ 1.3		$\langle \mathbf{u}_7, \mathbf{f}_1 \rangle$ 1.1	
$\langle \mathbf{u}_1, \mathbf{f}_2 \rangle$ 3.2	$\langle \mathbf{u}_2, \mathbf{f}_2 \rangle$ 5.2	....	$\langle \mathbf{u}_7, \mathbf{f}_2 \rangle$ 8.5	**max**
$\langle \mathbf{u}_1, \mathbf{f}_3 \rangle$ 5.1	$\langle \mathbf{u}_2, \mathbf{f}_3 \rangle$ -0.9		$\langle \mathbf{u}_7, \mathbf{f}_3 \rangle$ 4.6	

$$p_j = \mathrm{ReLU}(\max_i F_{ij})$$

for the video **n't rent it**

$\langle u_2, f_1 \rangle$ $\langle u_7, f_2 \rangle$

1.3 **8.5**

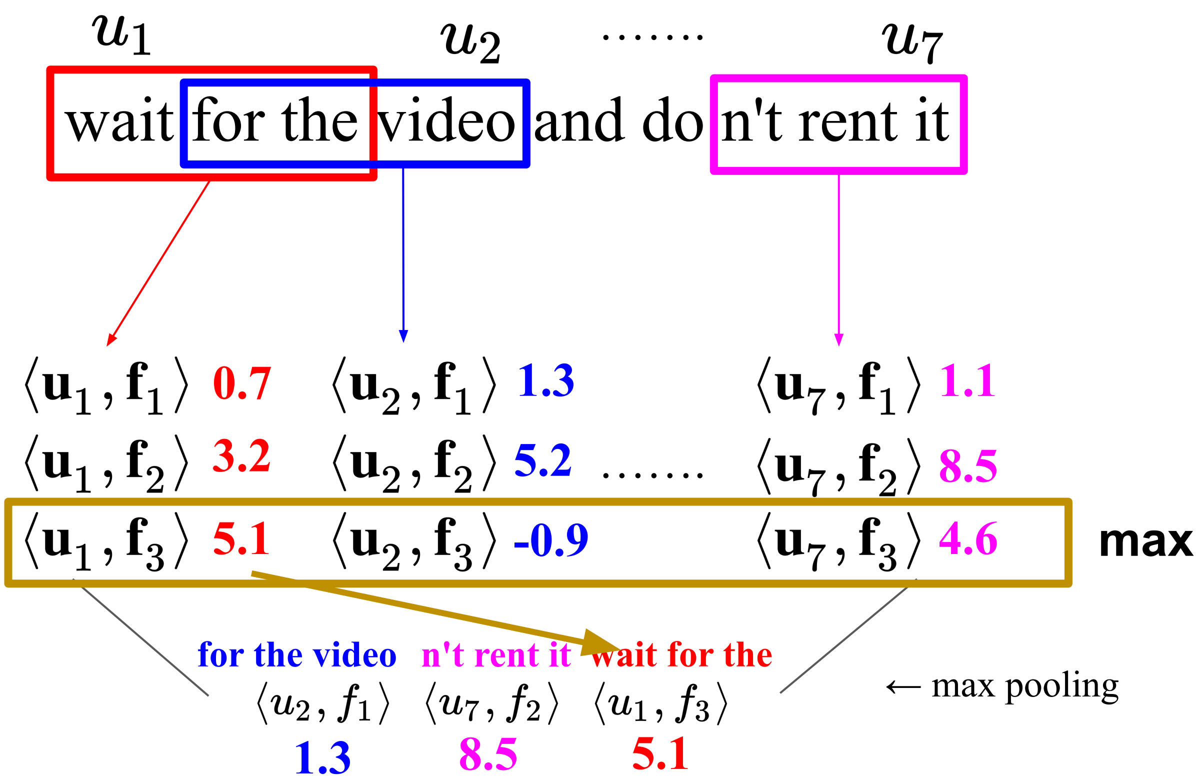
← max pooling

(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \leq n - \ell$$

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$$p_j = \mathrm{ReLU}(\max_i F_{ij})$$



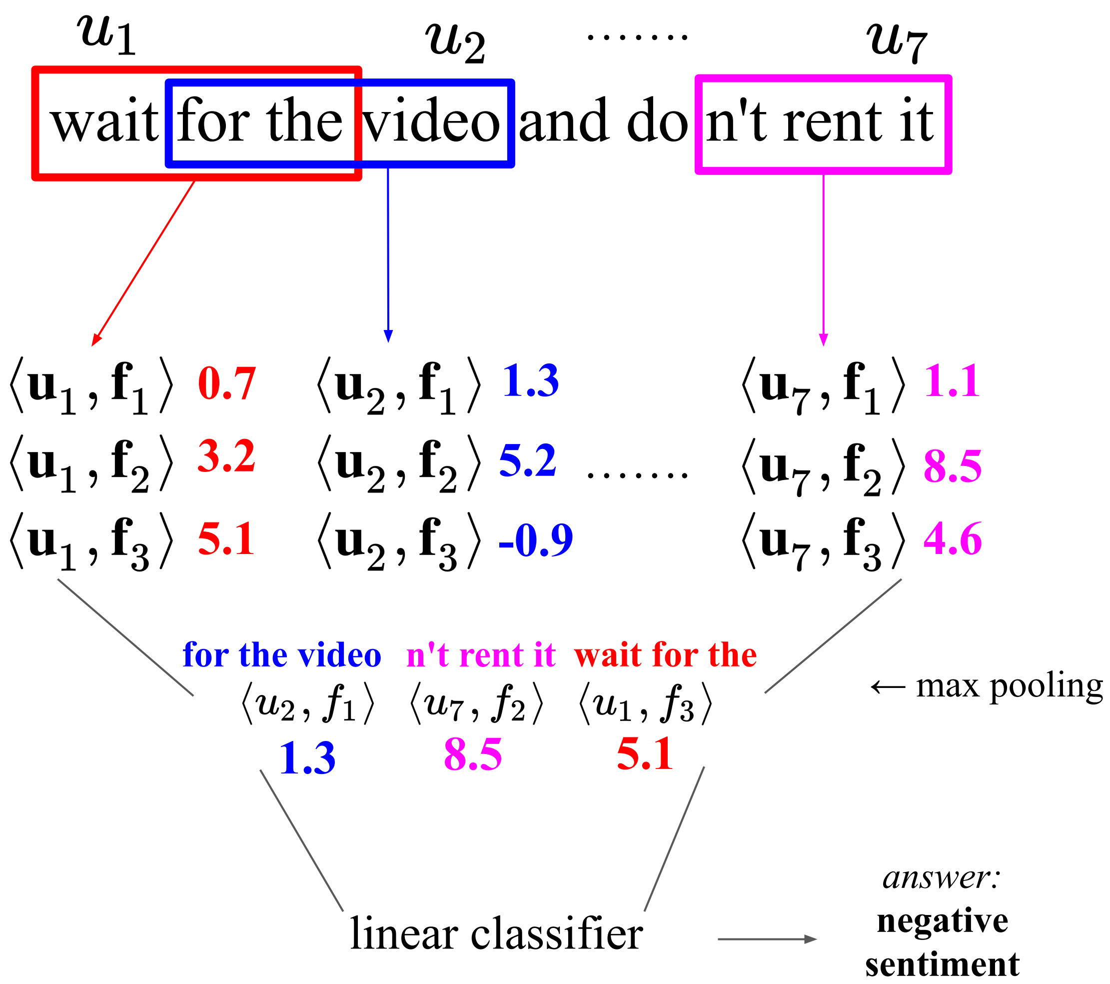
(word embeddings)

$$\mathbf{u}_i = [\mathbf{w}_i; \ldots; \mathbf{w}_{i+\ell-1}]$$
$$i \le n - \ell$$

$$F_{ij} = \langle \mathbf{u}_i, \mathbf{f}_j \rangle$$

$$p_j = \mathrm{ReLU}(\max_i F_{ij})$$

$$\mathbf{o} = \mathrm{softmax}(\mathbf{W}\mathbf{p})$$



Question 1

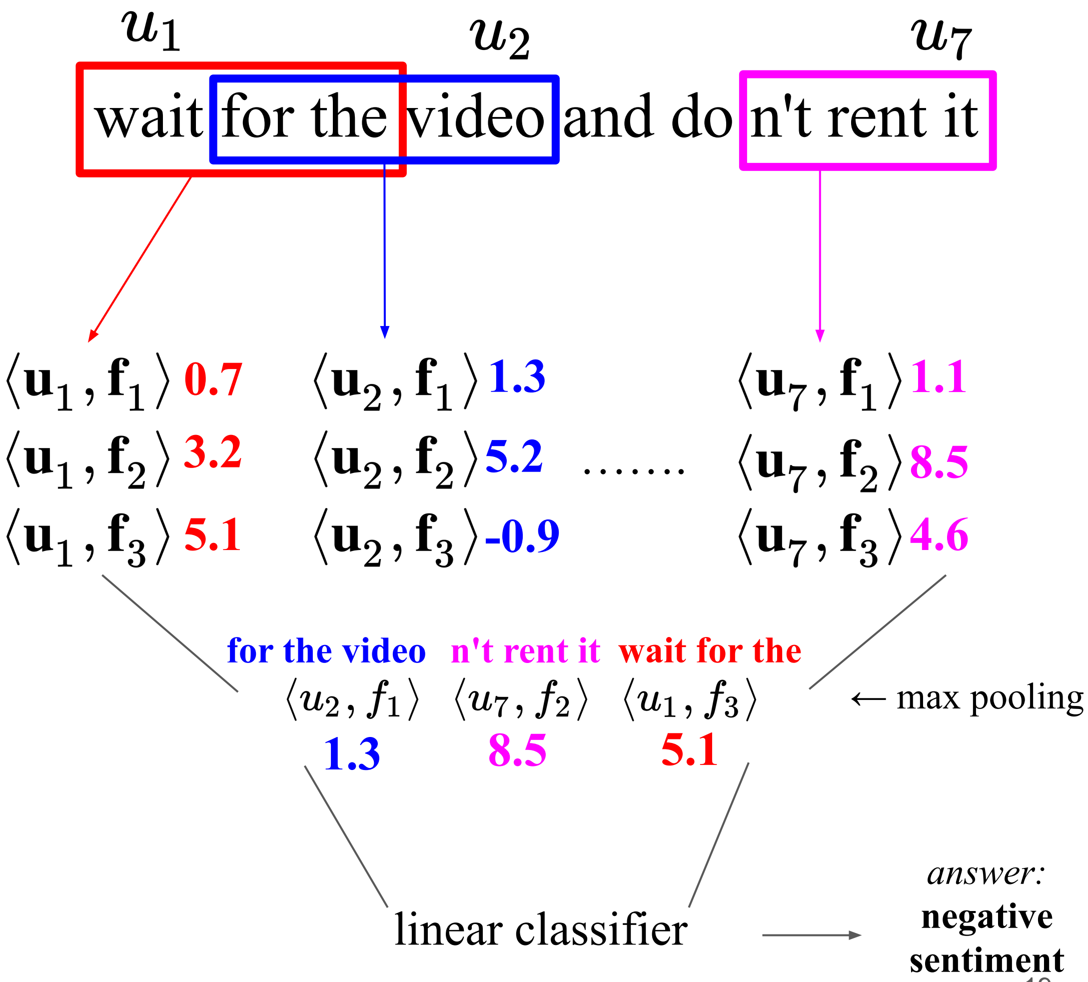
Which ngrams contribute to classification?

(*prediction interpretability*)

Which ngrams contribute to classification?

Common wisdom 1:

All of the ngrams that pass the max-pooling are informative.



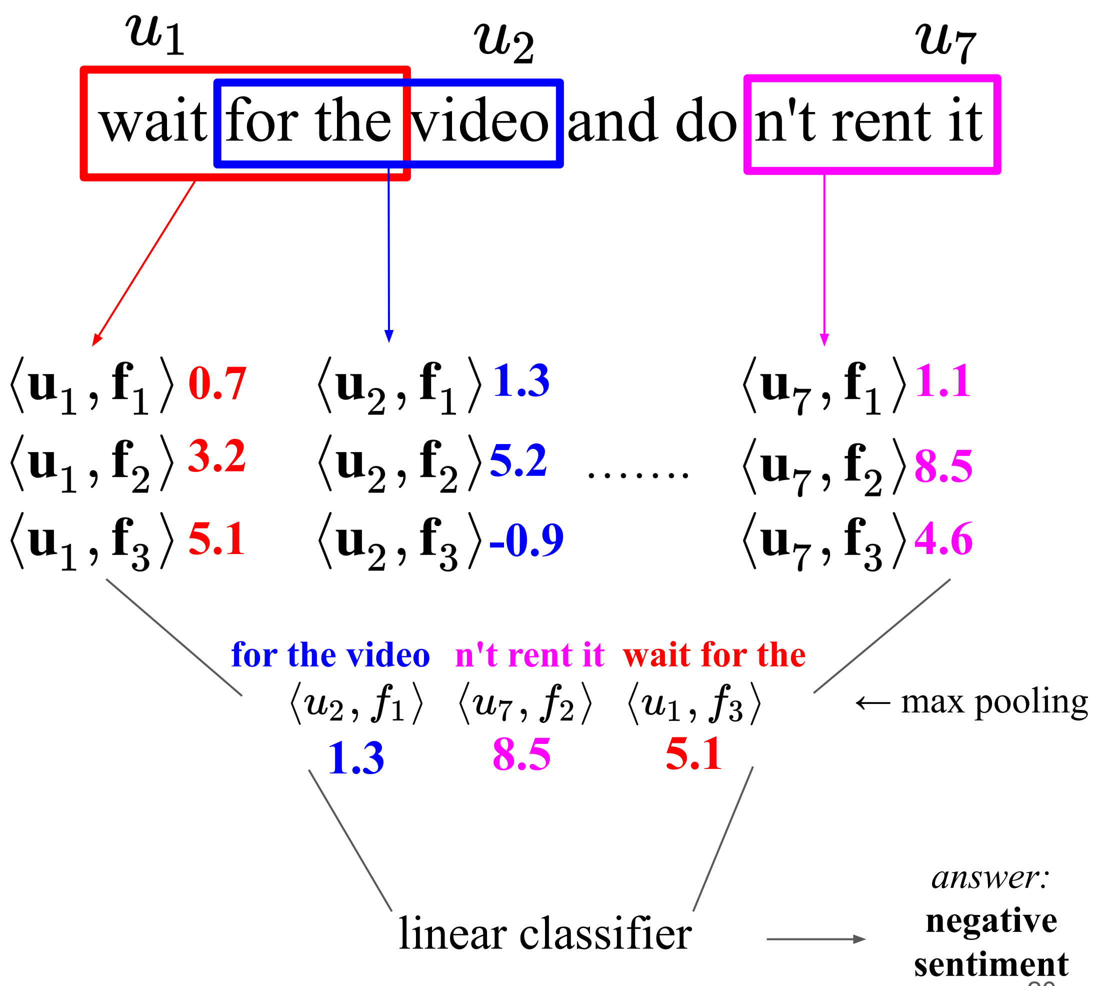
Which ngrams contribute to classification?

Common wisdom 1:

All of the ngrams that pass the max-pooling are informative.

But every filter **has to** choose an ngram, for every prediction.

Are all of them actually important?



Which ngrams contribute to classification?

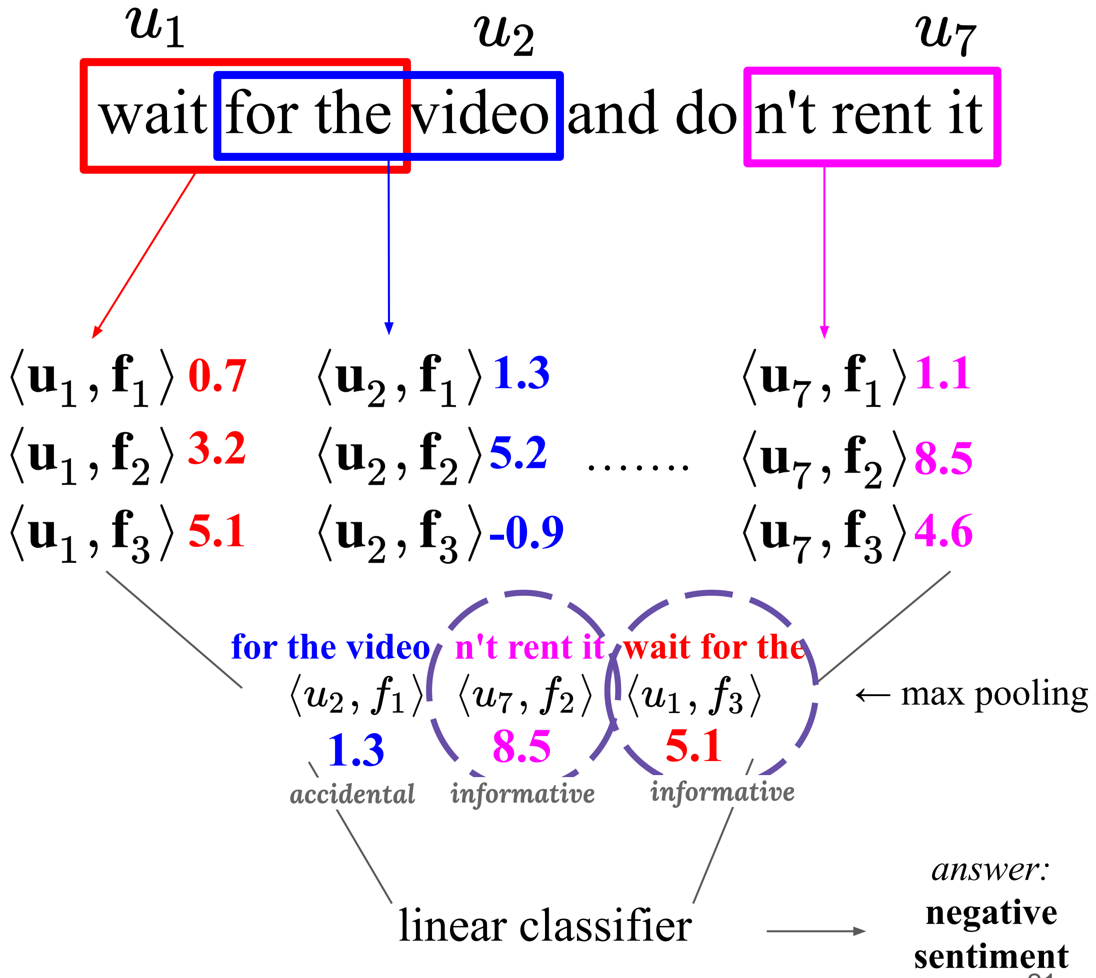
Common wisdom 1:

All of the ngrams that pass the max-pooling are informative.

But every filter **has to** choose an ngram, for every prediction.

Are all of them actually important?

Are some of them accidental?



Which ngrams contribute to classification

- We show how to differentiate between accidental vs informative ngrams
  - by fitting a 1d classifier
- We can remove 45% of pooled ngrams without hurting performance
  - → those 45% of discarded ngrams were not important

Details in the paper

Better prediction interpretability

input:
this product sucked was not loud at all lights did n't work overall a bad product that 's UNK taking up space
model answer: **negative**

Current tools supply explanations as the set of ngrams chosen in the max-pooling.

Better prediction interpretability

input:
this product sucked was not loud at all lights did n't work overall a bad product that 's UNK taking up space
model answer: **negative**

Current tools supply explanations as the set of ngrams chosen in the max-pooling.

Relevant ngrams:

filter	ngram
0	product sucked was
1	overall a bad
2	lights did n't
3	PAD this product
4	did n't work
5	sucked was not
6	work overall a
7	was not loud
8	a bad product
9	PAD PAD this

Better prediction interpretability

input:
this product **sucked was not** loud at all lights **did n't work** overall **a bad product** that 's UNK taking up space
model answer: **negative**

Current tools supply explanations as the set of ngrams chosen in the max-pooling.

By removing accidental ngrams, we can get a cleaner explanation.

Relevant ngrams:

filter	ngram
0	~~product sucked was~~
1	~~overall a bad~~
2	~~lights did n't~~
3	~~PAD this product~~
4	did n't work
5	sucked was not
6	~~work overall a~~
7	~~was not loud~~
8	a bad product
9	~~PAD PAD this~~

~~Common wisdom 1:~~

~~All of the ngrams that pass the max-pooling are informative.~~

Only **some** dimensions in the max-pooling output are informative.

We can find them.

Question 2

What does each filter capture?

(*model interpretability*)

What does each filter capture?

Common Wisdom 2:

Each filter captures a group of **closely-related** ngrams

f_1
- *had no issues*
- *had zero issues*
- *had no problems*

f_2
- *is super cool*
- *was very interesting*
- *are well beyond*

- 300 filters → 300 families of ngrams
- Each filter is *homogeneous* - captures one family.

Since we can check which ngrams are informative, we can verify if this is true.

We find a more complicated story

What is captured by a filter?

Assumption (more in paper): ngrams that have high activation represent the filter

Let's look at the ngrams that maximize a given filter.

$$\text{still working perfect} \longrightarrow \overset{f}{\begin{bmatrix}0.5\\2.1\\0.1\\3.4\\2.2\\2.5\end{bmatrix}} \longrightarrow 6.42$$

What is captured by a filter?

We can **decompose** this score into **slot scores** by dividing the inner product into word-level inner products:

$$\langle \mathbf{u}, \mathbf{f} \rangle = \sum_{i=0}^{\ell-1} \langle \mathbf{w_i}, \mathbf{f_{id:i(d+1)}} \rangle$$

$$f$$

$$\begin{array}{ccccc}
 & & \left[\begin{array}{c} 0.5 \\ 2.1 \\ \hline 0.1 \\ 3.4 \\ \hline 2.2 \\ 2.5 \end{array}\right] & & \\
\end{array}$$

$$\begin{array}{lcl}
\text{still} \rightarrow \begin{bmatrix} 0.5 \\ 2.1 \end{bmatrix} \rightarrow & 1.58 & \\
 & + & \\
\text{working} \rightarrow \begin{bmatrix} 0.1 \\ 3.4 \end{bmatrix} \rightarrow & 1.22 & = 6.42 \\
 & + & \\
\text{perfect} \rightarrow \begin{bmatrix} 2.2 \\ 2.5 \end{bmatrix} \rightarrow & 3.62 &
\end{array}$$

What is captured by a filter?

$$
\begin{array}{ccccc}
 & & f & & \\
\text{still} & \rightarrow & \left[\begin{array}{c} 0.5 \\ 2.1 \\ \hline 0.1 \\ 3.4 \\ \hline 2.2 \\ 2.5 \end{array}\right] & \begin{array}{c} \rightarrow \\ \\ \rightarrow \\ \\ \rightarrow \end{array} & \begin{array}{c} 1.58 \\ + \\ 1.22 \\ + \\ 3.62 \end{array} = 6.42 \\
\text{working} & \rightarrow & & & \\
\text{perfect} & \rightarrow & & &
\end{array}
$$

What is captured by a filter?

$$\begin{array}{c} \text{still} \\ \text{working} \\ \text{perfect} \end{array} \rightarrow \overset{f}{\left[\begin{array}{c} 0.5 \\ 2.1 \\ \hline 0.1 \\ 3.4 \\ \hline 2.2 \\ 2.5 \end{array}\right]} \rightarrow \begin{array}{c} 1.58 \\ + \\ 1.22 \\ + \\ 3.62 \end{array} = 6.42$$

We can generate the ngrams that maximize each filter slot separately:

$$\begin{array}{c} \text{saves} \\ \text{delight} \\ \text{invaluable} \end{array} \rightarrow \overset{f}{\left[\begin{array}{c} 0.5 \\ 2.1 \\ \hline 0.1 \\ 3.4 \\ \hline 2.2 \\ 2.5 \end{array}\right]} \rightarrow \begin{array}{c} 2.52 \\ + \\ 2.29 \\ + \\ 4.19 \end{array} = 9.0$$

We observe an interesting phenomenon:

The generated maximized ngrams score **much higher** than the top ngrams.

filter	top ngram	score	top word for each slot	score
f1	poorly designed junk	7.31	poorly displaying landfill	**10.28**
f2	utterly useless .	6.33	stopped refund disabled	**7.96**
f3	still working perfect	6.42	saves delight invaluable	**9.0**
f4	a minor drawback	6.11	workstation high-quality drawback	**9.27**
f5	deserves four stars	5.56	excelente crossover incredible	**7.78**

We observe an interesting phenomenon:

The generated maximized ngrams score **much higher** than the top ngrams.

filter	top ngram	score	top word for each slot	score
f1	poorly designed junk	7.31	poorly displaying landfill	**10.28**
f2	utterly useless .	6.33	stopped refund disabled	**7.96**
f3	still working perfect	6.42	saves delight invaluable	**9.0**
f4	a minor drawback	6.11	workstation high-quality drawback	**9.27**
f5	deserves four stars	5.56	excelente crossover incredible	**7.78**

We observe an interesting phenomenon:

The generated maximized ngrams score **much higher** than the top ngrams.

Why?

filter	top ngram			score
f1	poorly designe			**10.28**
f2	utterly useless			**7.96**
f3	still working pe			**9.0**
f4	a minor drawback	6.11	workstation high-quality drawback	**9.27**
f5	deserves four stars	5.56	excelente crossover incredible	**7.78**

What is captured by a filter?

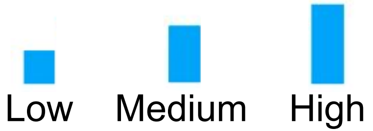
Let's look at the top ngrams for a specific filter:

rank	top ngrams				
	ngram	score	slot scores		
1	still working **perfect**	6.42	1.58	1.22	**3.62**
2	**works** - **perfect**	5.78	**1.91**	0.25	**3.62**
3	isolation proves **invaluable**	5.61	0.39	1.03	**4.19**
4	still near **perfect**	5.6	1.58	0.4	**3.62**
5	still working great	5.45	1.58	1.22	2.65
6	**works** as good	5.44	**1.91**	1.45	2.08
7	still **holding** strong	5.37	1.58	**1.81**	1.98

Only some of the words maximize their slot's score.

(they are in bold)

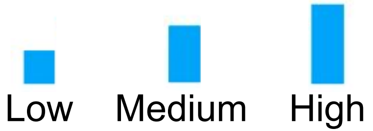
New concept: Slot Activation Pattern



List of top-scoring ngrams for a specific filter

ngram	slot #1	slot #2	slot #3
was super intriguing	1.01	3.16	5.84
go wrong pairing	3.97	4.12	1.65
am so grateful	2.59	3.27	4.07
overall very worth	3.84	1.86	4.22
go wrong bringing	3.97	4.12	1.81
also well worth	1.83	3.06	4.22
- super compassionate	0.51	3.17	5.01
go wrong when	3.97	4.12	-0.4
a well oiled	0.75	3.06	4.84

New concept: Slot Activation Pattern



List of top-scoring ngrams for a specific filter

ngram	slot #1	slot #2	slot #3
was super intriguing	1.01	3.16	5.84
go wrong pairing	3.97	4.12	1.65
am so grateful	2.59	3.27	4.07
overall very worth	3.84	1.86	4.22
go wrong bringing	3.97	4.12	1.81
also well worth	1.83	3.06	4.22
- super compassionate	0.51	3.17	5.01
go wrong when	3.97	4.12	-0.4
a well oiled	0.75	3.06	4.84

New concept: Slot Activation Pattern

List of top-scoring ngrams for a specific filter

ngram	slot #1	slot #2	slot #3
was super intriguing	1.01	3.16	5.84
go wrong pairing	3.97	4.12	1.65
am so grateful	2.59	3.27	4.07
overall very worth	3.84	1.86	4.22
go wrong bringing	3.97	4.12	1.81
also well worth	1.83	3.06	4.22
- super compassionate	0.51	3.17	5.01
go wrong when	3.97	4.12	-0.4
a well oiled	0.75	3.06	4.84

New concept: Slot Activation Pattern

ngram	slot #1	slot #2	slot #3
was super intriguing	1.01	3.16	5.84
go wrong pairing	3.97	4.12	1.65
am so grateful	2.59	3.27	4.07
overall very worth	3.84	1.86	4.22
go wrong bringing	3.97	4.12	1.81
also well worth	1.83	3.06	4.22
- super compassionate	0.51	3.17	5.01
go wrong when	3.97	4.12	-0.4
a well oiled	0.75	3.06	4.84

Cluster filter ngrams according to slot activations

Each cluster is a homogeneous family of ngrams.

The same filter detected both families.

cluster 1

cluster 2

ngram	slot #1	slot #2	slot #3
centroid	0.75	1.97	2.79
was super intriguing	1.01	3.16	5.84
am so grateful	2.59	3.27	4.07
overall very worth	3.84	1.86	4.22
also well worth	1.83	3.06	4.22
- super compassionate	0.51	3.17	5.01
a well oiled	0.75	3.06	4.84
centroid	2.87	2.17	0.12
go wrong bringing	3.97	4.12	1.81
go wrong pairing	3.97	4.12	1.65
go wrong when	3.97	4.12	-0.4

Finding (i): Filters are not homogeneous

Each filter detects multiple distinct families of ngrams.

Validated by using clustering on the slot vectors.

ngram	slot #1	slot #2	slot #3
centroid	0.75	1.97	2.79
was super intriguing	1.01	3.16	5.84
am so grateful	2.59	3.27	4.07
overall very worth	3.84	1.86	4.22
also well worth	1.83	3.06	4.22
- super compassionate	0.51	3.17	5.01
a well oiled	0.75	3.06	4.84
centroid	2.87	2.17	0.12
go wrong bringing	3.97	4.12	1.81
go wrong pairing	3.97	4.12	1.65
go wrong when	3.97	4.12	-0.4

What does each filter capture?

~~Common Wisdom 2:~~

~~Each filter captures a group of closely-related ngrams~~

- ~~300 filters → 300 families of ngrams~~
- ~~Each filter is *homogenous*~~

Filters can be *heterogeneous*

300 filters → ? (>300) families of ngrams

What does each filter capture?

Common Wisdom 3:

Each filter detects the existence of ngrams.

In other words: each slot position detects the existence of specific words.

What does each filter capture?

Common Wisdom 3:

Each filter detects the existence of ngrams.

In other words: each slot position detects the existence of specific words.

2.59	1.86	5.05
'm	really	pleased

High-scoring ngram

does slot #2 capture the word "*really*"?

What does each filter capture?

Common Wisdom 3:

Each filter detects the existence of ngrams.

In other words: each slot position detects the existence of specific words.

weak score

2.59 ~1.86 5.05

'm ____ pleased

High-scoring ngram

1.86 is an average score for slot #2.
many words get similar scores

What does each filter capture?

Common Wisdom 3:

Each filter detects the existence of ngrams.

In other words: each slot position detects the existence of specific words.

weak score

2.59 ~1.86 5.05

'm ____ pleased

High-scoring ngram

Hypothesis: these weak scores indicate a **wildcard**.
Is it really?

Finding (ii): Negative Ngrams

weak score

2.59 ~1.86 5.05

'm ____ pleased

High-scoring ngram

strong negative score

-3.4

'm not pleased

Low-scoring ngram

slot #2 is detecting the **absence of** the word "*not*"

Finding (ii): Negative Ngrams

A weak slot may signify that instead of detecting words, it detects *the lack of* words.

We can search for variants of high-scoring ngrams that are low-scoring.

weak score

2.59 ~1.86 5.05

'm ____ pleased

strong negative score

-3.4

'm not pleased

Negative ngram

slot #2 is detecting the **absence of** the word "*not*"

Better prediction interpretability (2)

By

- decomposing the ngram score to word-scores
- highlighting negative ngrams,

we can improve the explanation

Better prediction interpretability (2)

By

- decomposing the ngram score to word-scores
- highlighting negative ngrams,

we can improve the explanation

this product **sucked was not** loud at all lights **did n't work** *overall* ***a bad*** **product** that 's UNK taking up space

filter	ngram	slot scores		
4	did n't work	1.21	0.97	2.65
5	sucked was not	0.98	0.59	1.32
8	a bad product	-0.45	3.08	1.32

Better prediction interpretability (2)

negative
ngram

By

- decomposing the ngram score to word-scores
- highlighting negative ngrams,

we can improve the explanation

this product **sucked was not** loud at all lights **did n't work** *overall* ***a bad*** **product** that 's UNK taking up space

filter	ngram	slot scores		
1	*overall a bad*	*2.53*	*1.4*	*-1.16*
4	did n't work	1.21	0.97	2.65
5	sucked was not	0.98	0.59	1.32
8	a bad product	-0.45	3.08	1.32

Better prediction interpretability (2)

negative ngram

By

- decomposing the ngram score to word-scores
- highlighting negative ngrams,

we can improve the explanation

this product **sucked was not** loud at all lights **did n't work** *overall* ***a bad*** **product** that 's UNK taking up space

filter	ngram	slot scores		
1	*overall a bad*	*2.53*	*1.4*	*-1.16*
4	did n't work	1.21	0.97	2.65
5	sucked was not	0.98	0.59	1.32
8	a bad product	-0.45	3.08	1.32

Model Interpretability

The work shown thus far can be classified as *model interpretability*, or explaining a given model as a whole.

By interpreting filters directly we can better understand the model's captured functionality:

- assign sets (*clusters*) of representative ngrams to each filter
- highlight "positive slots" and "negative slots"

Conclusion

Two main contributions

- Max-pooling induces classifying behavior
  - separates informative from non-informative features
  - implications beyond CNNs or text classification
- Filters in CNN text classification are not homogeneous
  - rely on activation patterns to capture different families of ngrams

Additionally, we present the tools to derive the informative ngram classes for each filter, **improving model and prediction interpretability**.