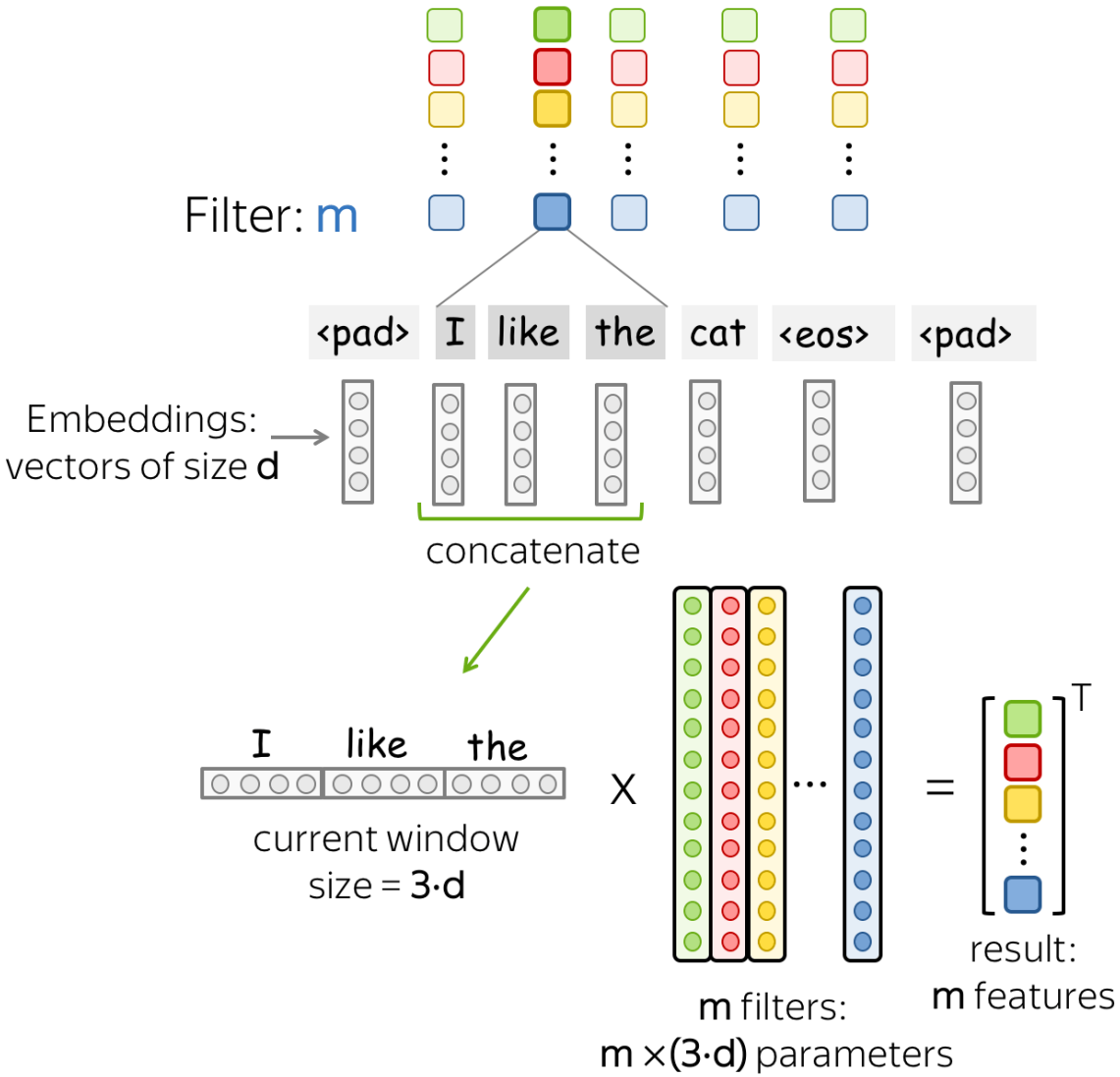
Foundations for Natural Language Processing

Recurrent Neural Networks (RNNs): Classification and Language Modeling

Ivan Titov
(with graphics/materials from Elena Voita)

School of informatics

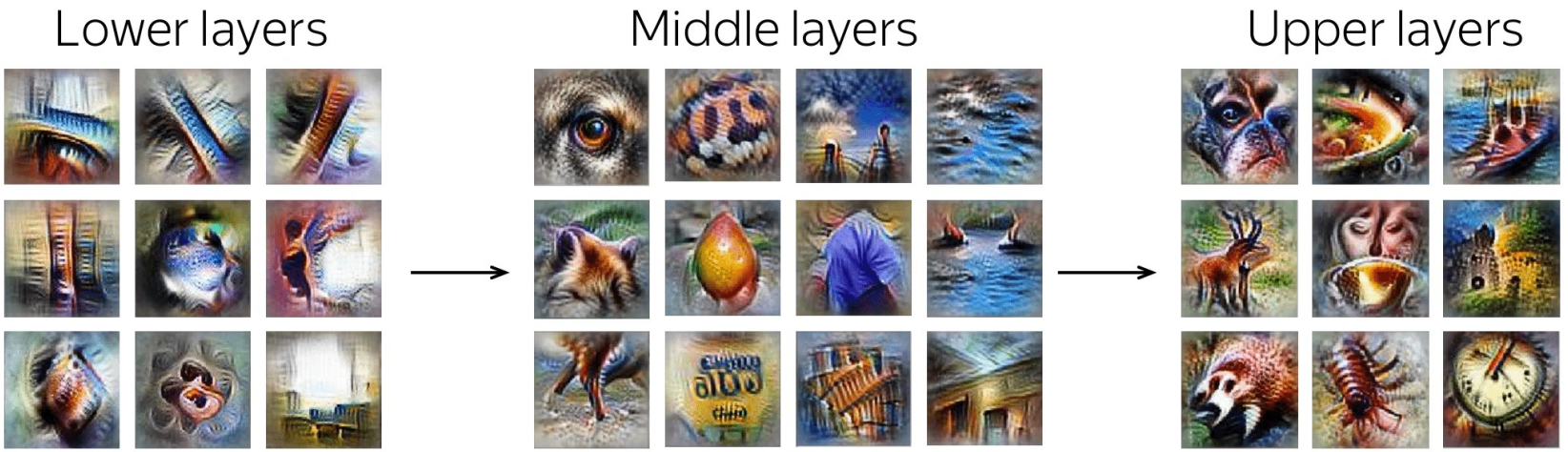
Recap: Convolution layer for text



Interpretability: what are CNNs learning?

It is informative / interesting to understand what individual CNN filters capture, in different layers

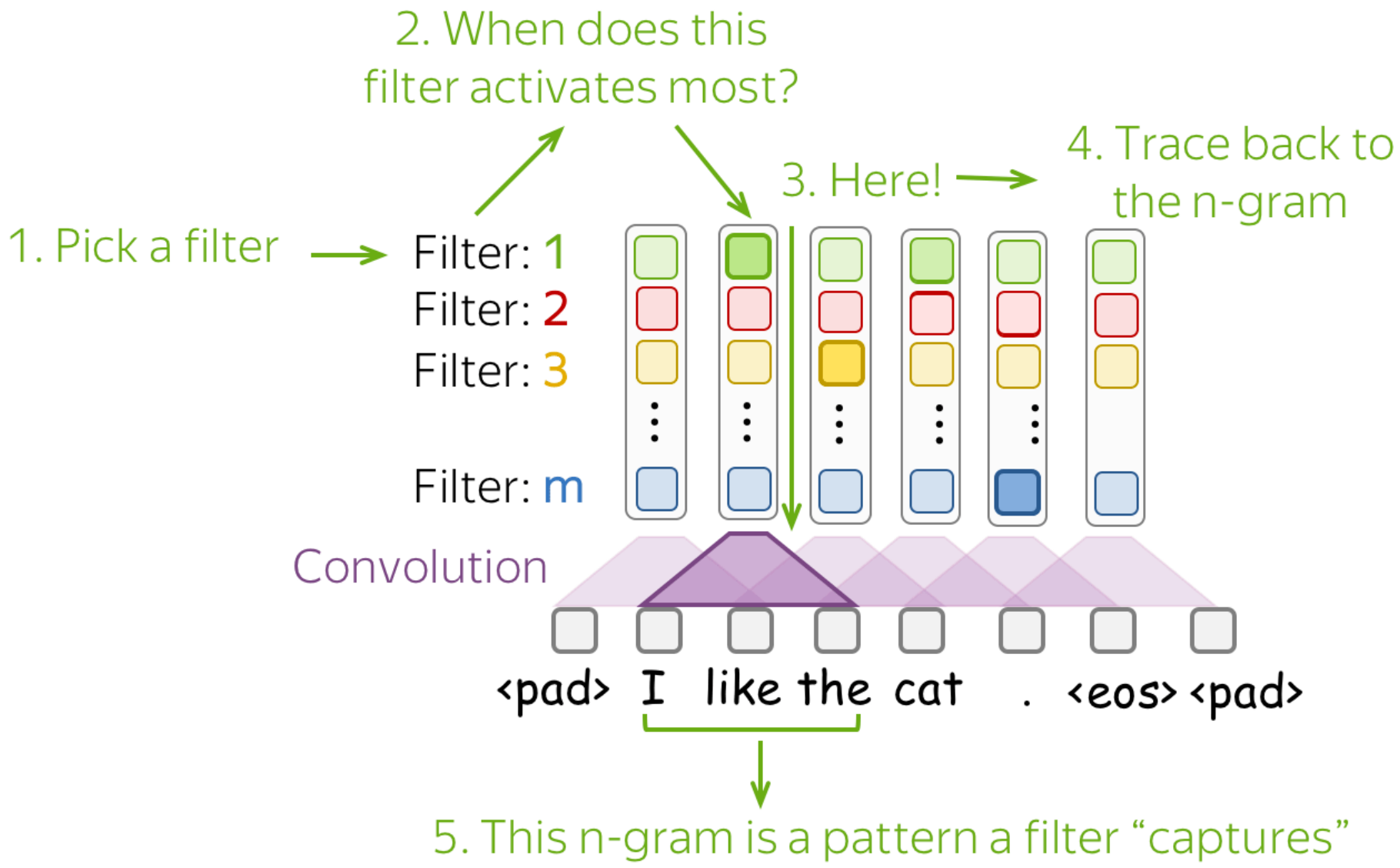
CNN filters in image processing models:



Can we get something like this for NLP models?

from https://distill.pub/2019/activation-atlas/

Interpretability: what are CNNs learning?



Filter activations

filter	Top n-gram	Score
1	poorly designed junk	7.31
2	simply would not	5.75
3	a minor drawback	6.11
4	still working perfect	6.42
5	absolutely gorgeous .	5.36
6	one little hitch	5.72
7	utterly useless .	6.33
8	deserves four stars	5.56
9	a mediocre product	6.91

Top n-grams for filter 4		Score
1	still working perfect	6.42
2	works - perfect	5.78
3	isolation proves invaluable	5.61
4	still near perfect	5.6
5	still working great	5.45
6	works as good	5.44
7	still holding strong	5.37

A filter activates for a family of n-grams with similar meaning

You can draw parallels with logistic regressions, relying on ngrams
What are the key differences?

Summary for Neural Text Classification (so far)

We considered

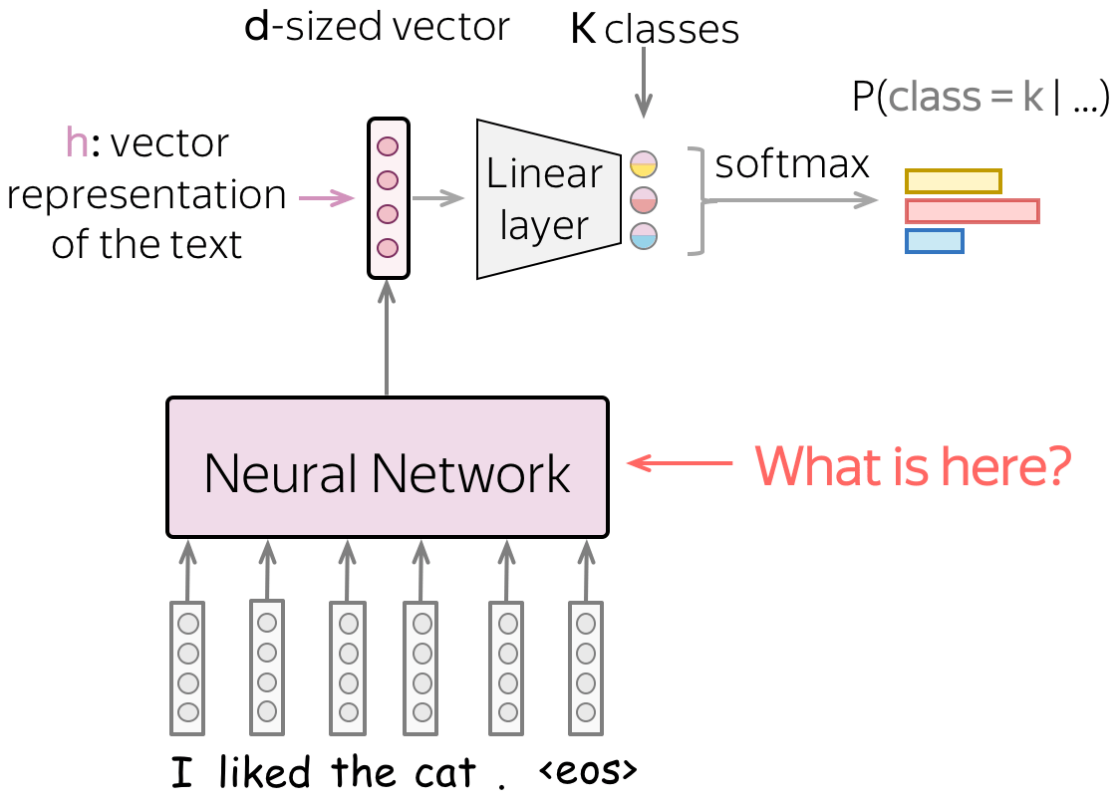
- Generalization of logistic regression
- Easy to integrate embeddings, estimated on unlabeled text
- BoW models, weighted BoW models, CNNs

Not exactly true for multilayered CNNs

All these models model only limited interaction between nonadjacent expressions, how do we handle these?

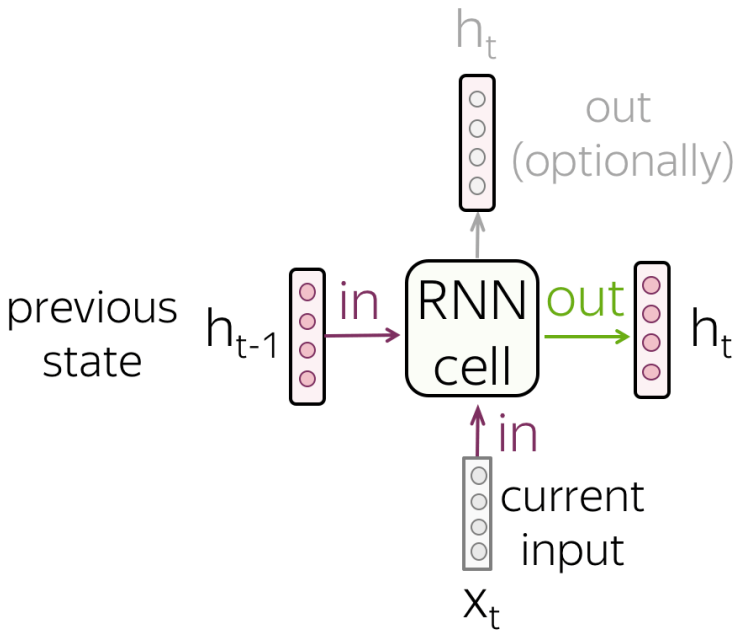
Now - Recurrent Neural Networks (RNNs)
Next week (not next time – on Friday) - Transformer models

Recap: NNs for text classification

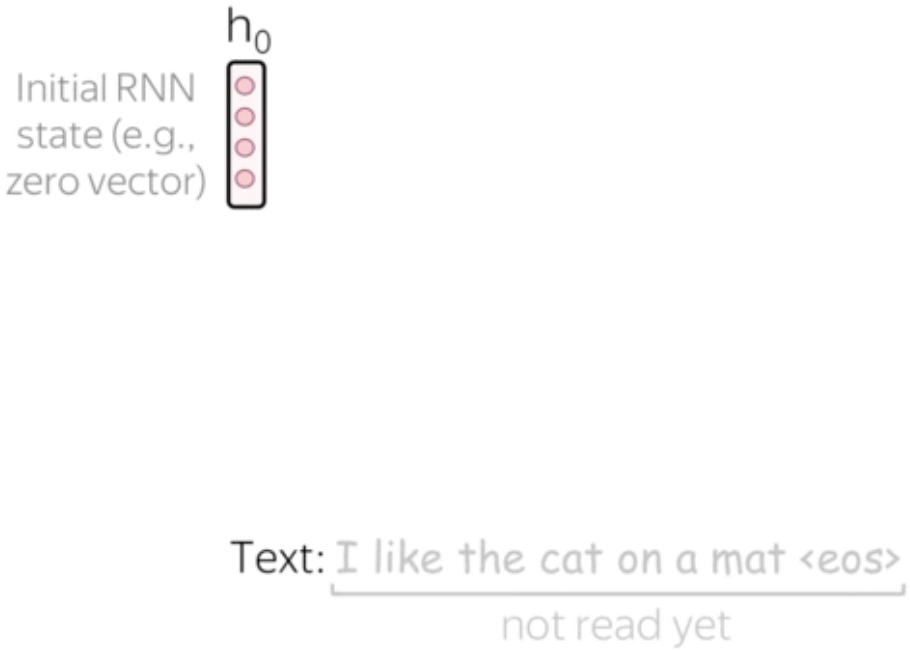


We will finish up with classification today by introducing a class of models which will be particularly useful for text generation (RNNs)

Recurrent Neural Networks: RNN cell



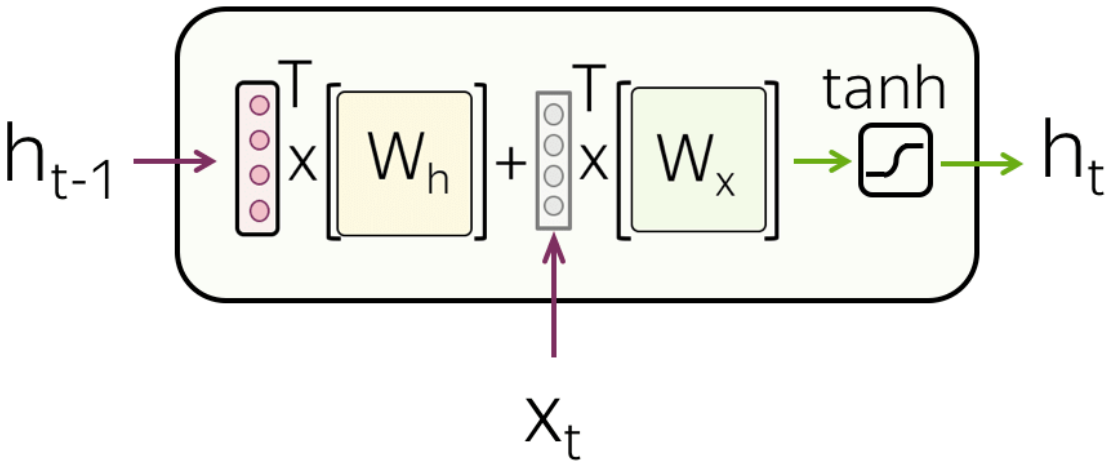
RNN reads a sequence of tokens



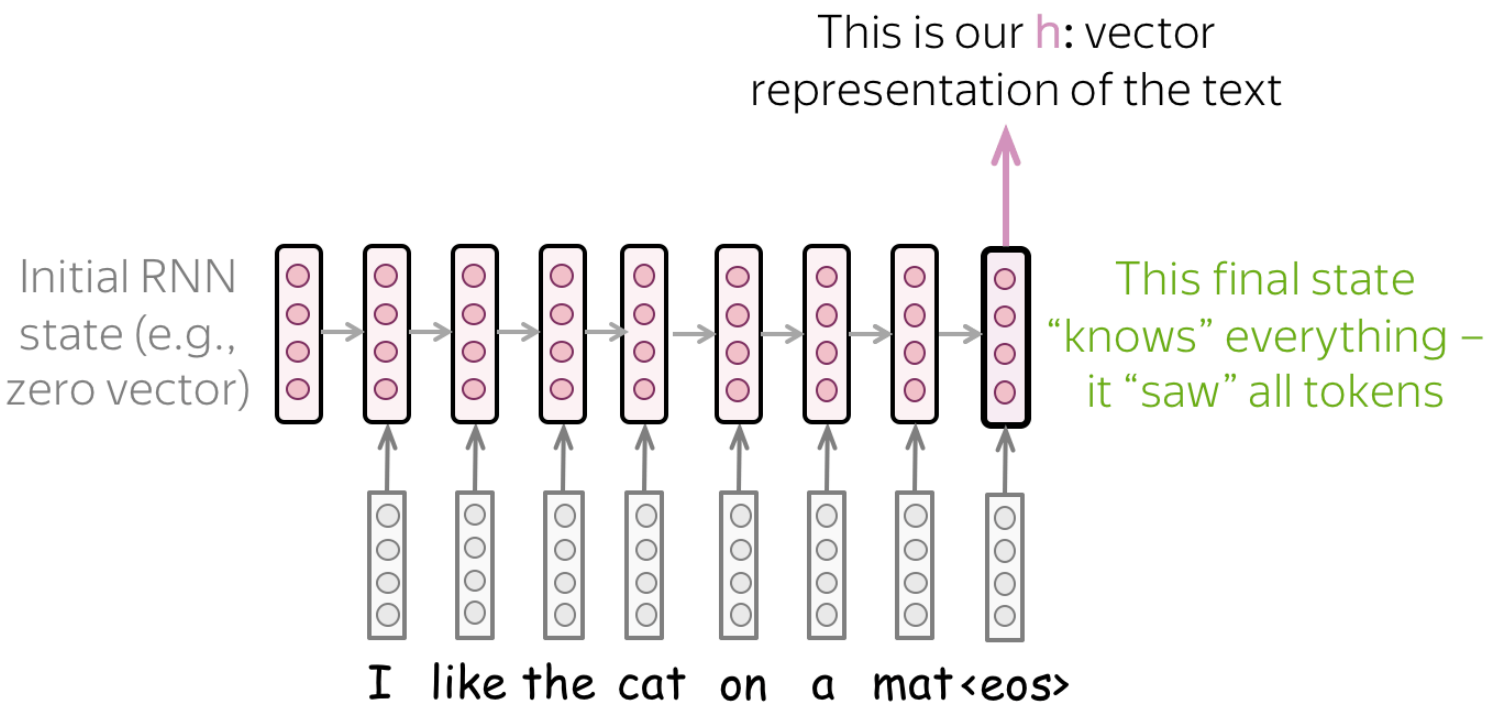
(video, not visible in pdf)

Vanilla RNN

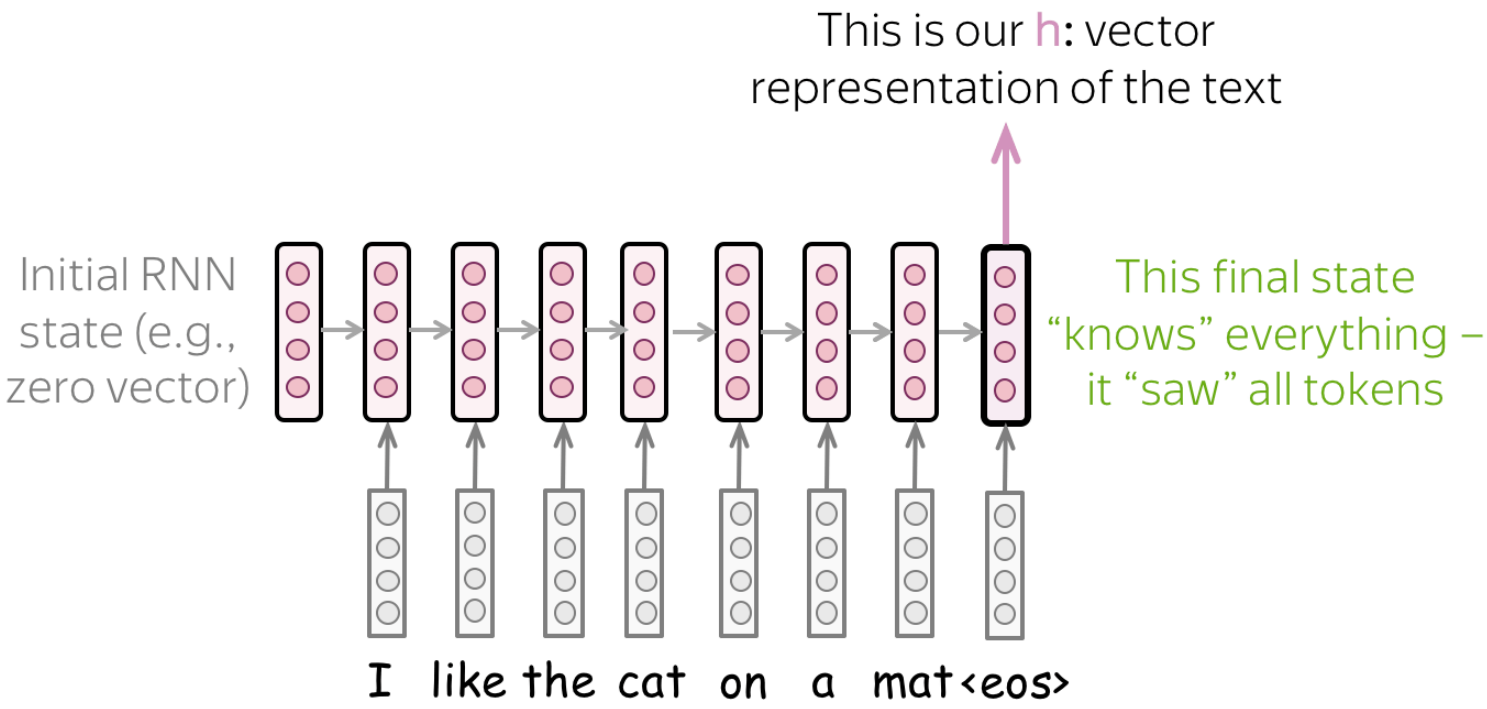
$$h_t = \tanh(h_{t-1}W_h + x_t W_x)$$



Text representation with RNN



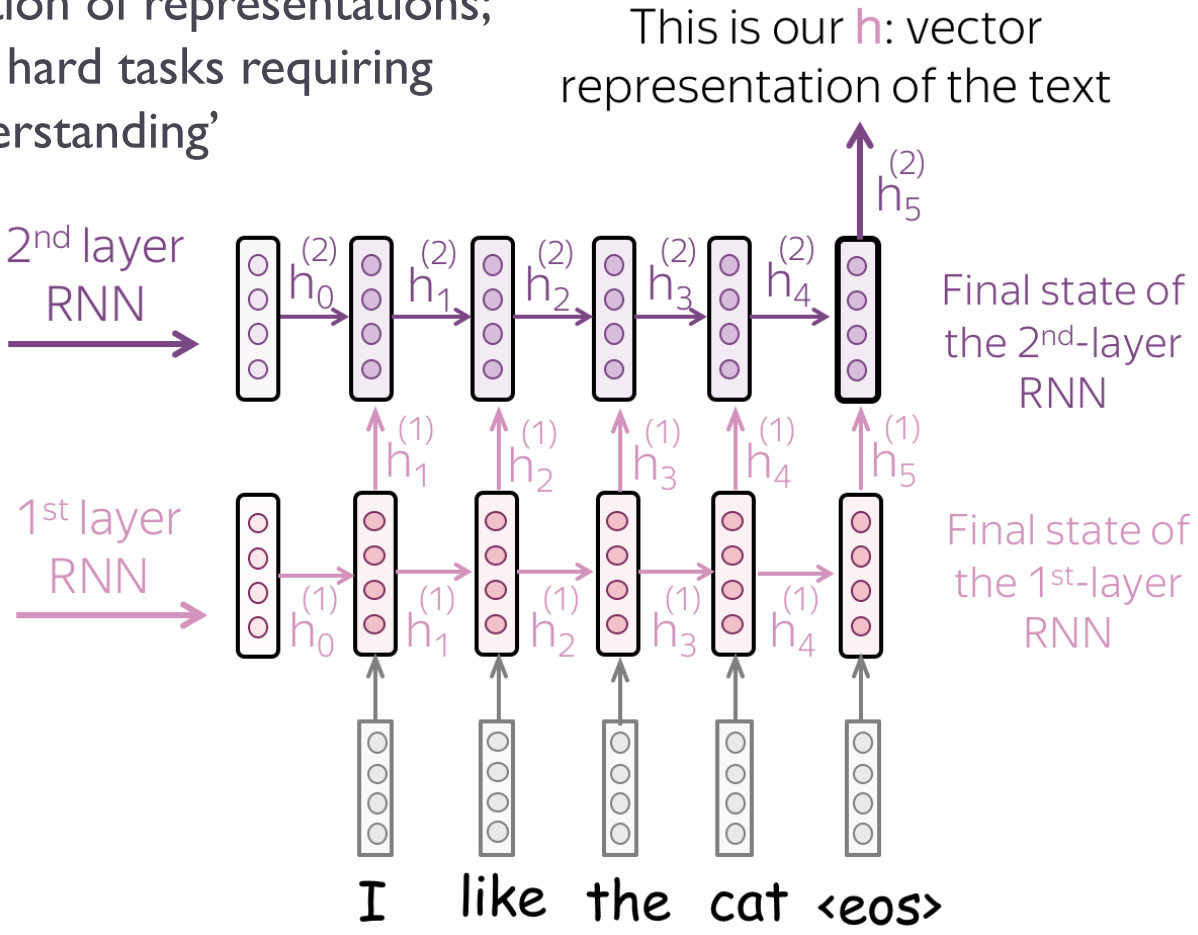
Text representation with RNN



The architecture may not be expressive enough, how can we make the model more powerful?

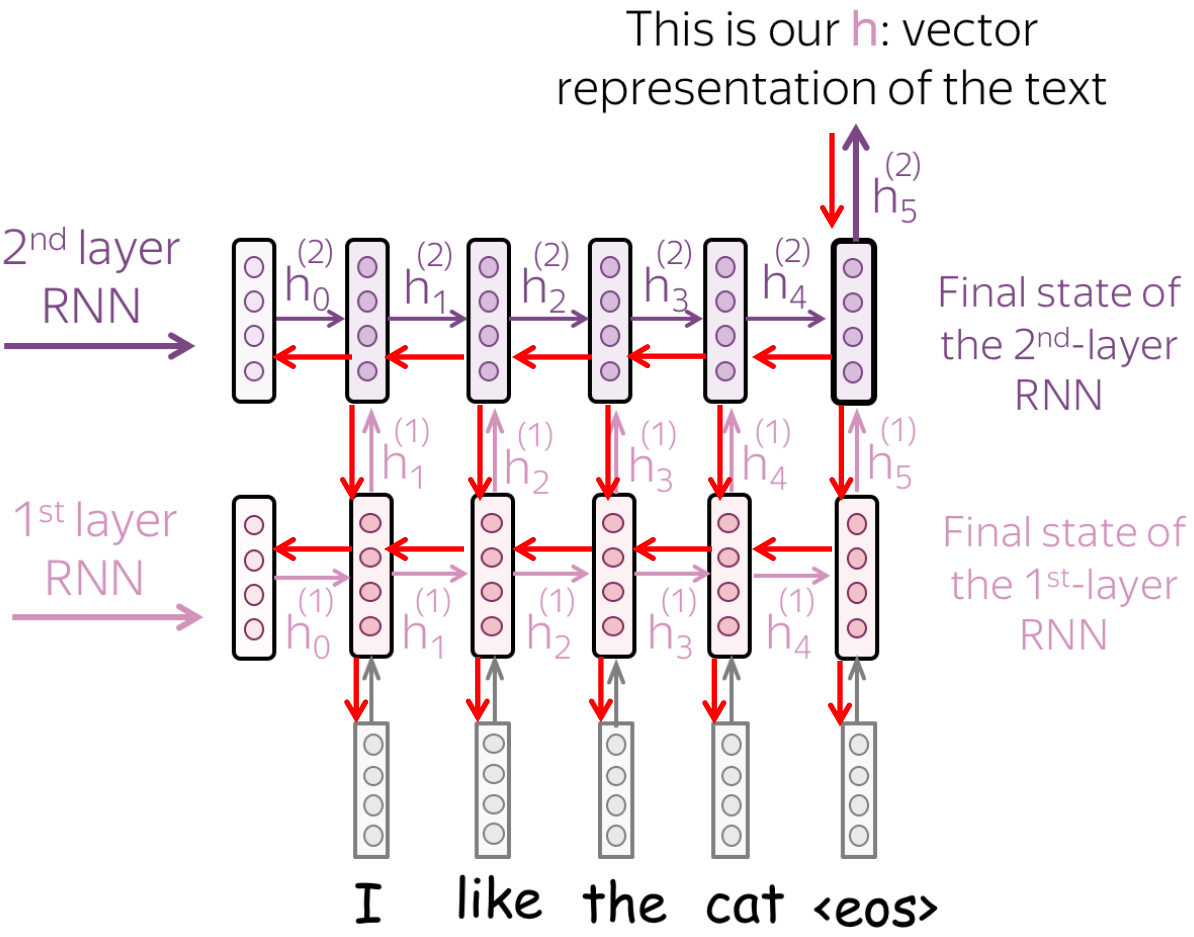
Text representation with multi-layer RNN

Better at capturing different levels of abstraction of representations; crucial for hard tasks requiring 'deep understanding'



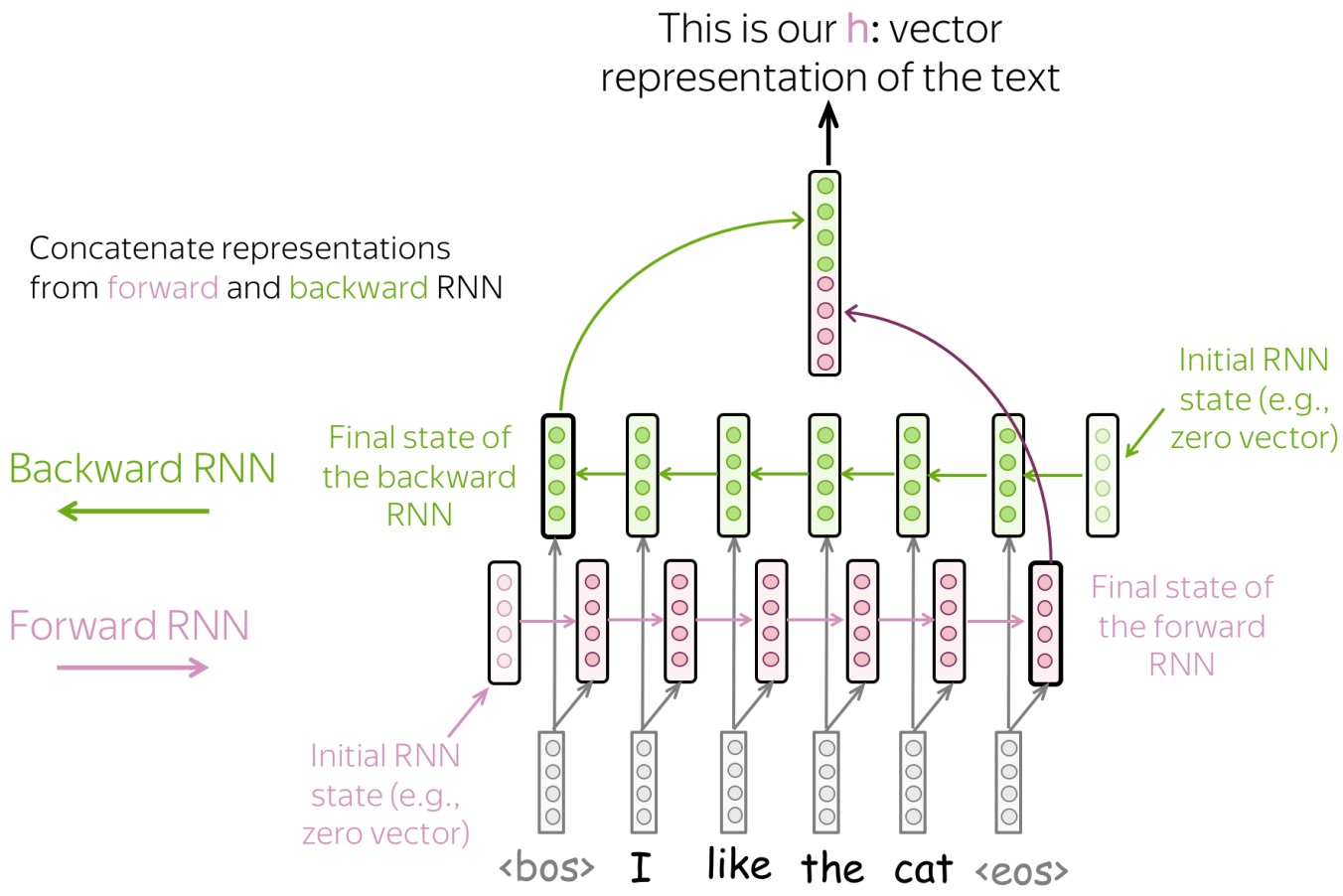
Is there a problem with passing only the last state as *h*?

Text representation with multi-layer RNN



Models learn by backpropagation, it takes many steps to propagate to the very beginning of the sentence; the model will not learn to reliably encode early parts of the sentence, how can we address it?

Text representation with bidirectional RNN

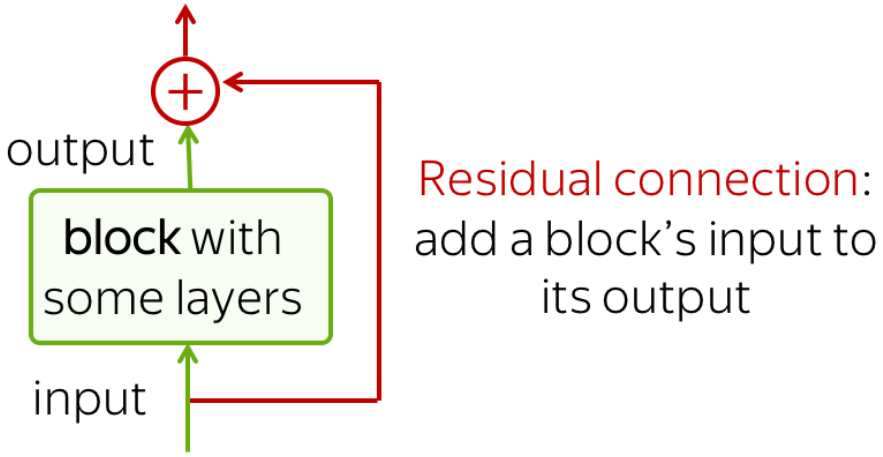


Stacking many layers

Unfortunately, when stacking a lot of layers, you can have a problem with propagating gradients from top to bottom through a deep network.

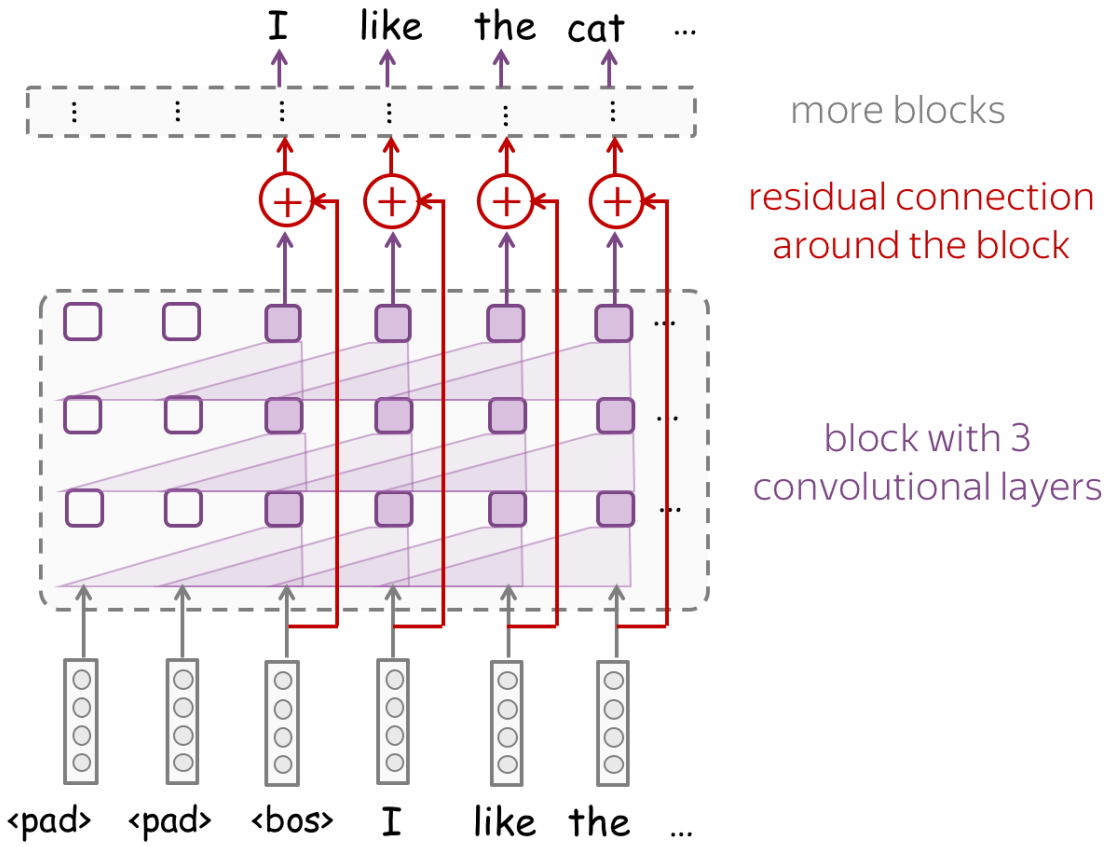
To mitigate, this we use Residual or Highway connections

Residual connections:



Multilayer models with residual connections

This is an example of multilayer CNN but it would look the same with multilayer / bi-directional RNNs



Having residual connection is necessary for training large-language models typically powered with Transformers (will talk about them in 3 lectures)

Summary on classification

- Naïve Bayes

  very fast to train, robust, makes overly strong assumptions

- Logistic regression

  still easy to train, requires strong features, fewer assumptions

- Convolutional Neural Networks

  still easy to train, fewer assumptions, but still (~) breaks sentence into ngrams

- Recurrent Neural Networks

  does not break a sentence into pieces, but the information is carried within the sentence through a vector

Language modeling

Recall, the language model assigns the probability to a sequence of words $y_1, y_2, \ldots, y_n$, relying on the chain rule:

$$P(y_1, y_2, \ldots, y_n) = P(y_1) \cdot P(y_2|y_1) \cdot P(y_3|y_1, y_2) \cdot \cdots \cdot P(y_n|y_1, \ldots, y_{n-1})$$

$$= \prod_{t=1}^{n} P(y_t|y_{<t})$$

How do we compute $P(y_t|y_{<t})$?

Ngram models made independence assumptions, basically breaking the sequence into smaller subsequences for estimation

Samples from ngram models: any issues?

```
hahn , director of the christian " love and
compassion " was designed as a result of any form ,
in the transaction is active in the stuva grill .
_eos_
```

```
pupils from eastern europe , africa , saudi arabia
' s church , yearn for such an open structure of
tables several times on monday 14 september 2003 ,
his flesh when i was curious to know and also to
find what they are constructed with a speeding
arrow . _eos_
```

Samples from ngram models: any issues?

```
hahn , director of the christian " love and
compassion " was designed as a result of any form ,
in the transaction is active in the stuva grill .
_eos_
```

```
pupils from eastern europe , africa , saudi arabia
' s church , yearn for such an open structure of
tables several times on monday 14 september 2003 ,
his flesh when i was curious to know and also to
find what they are constructed with a speeding
arrow . _eos_
```

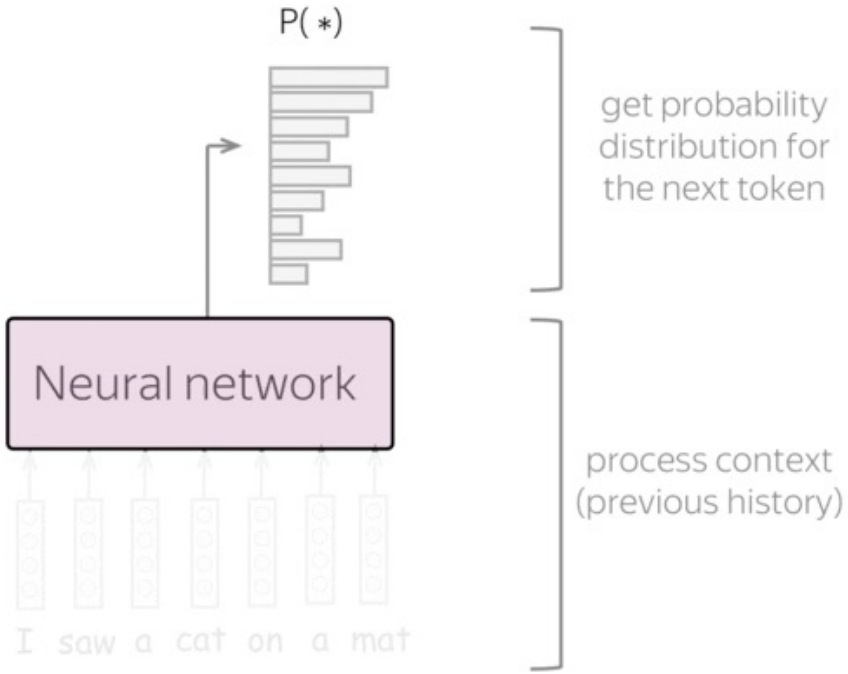
Ngram models clearly struggle with capturing longer context

NNs (e.g., RNNs) will let us model text without making explicit independence assumptions

Intuition

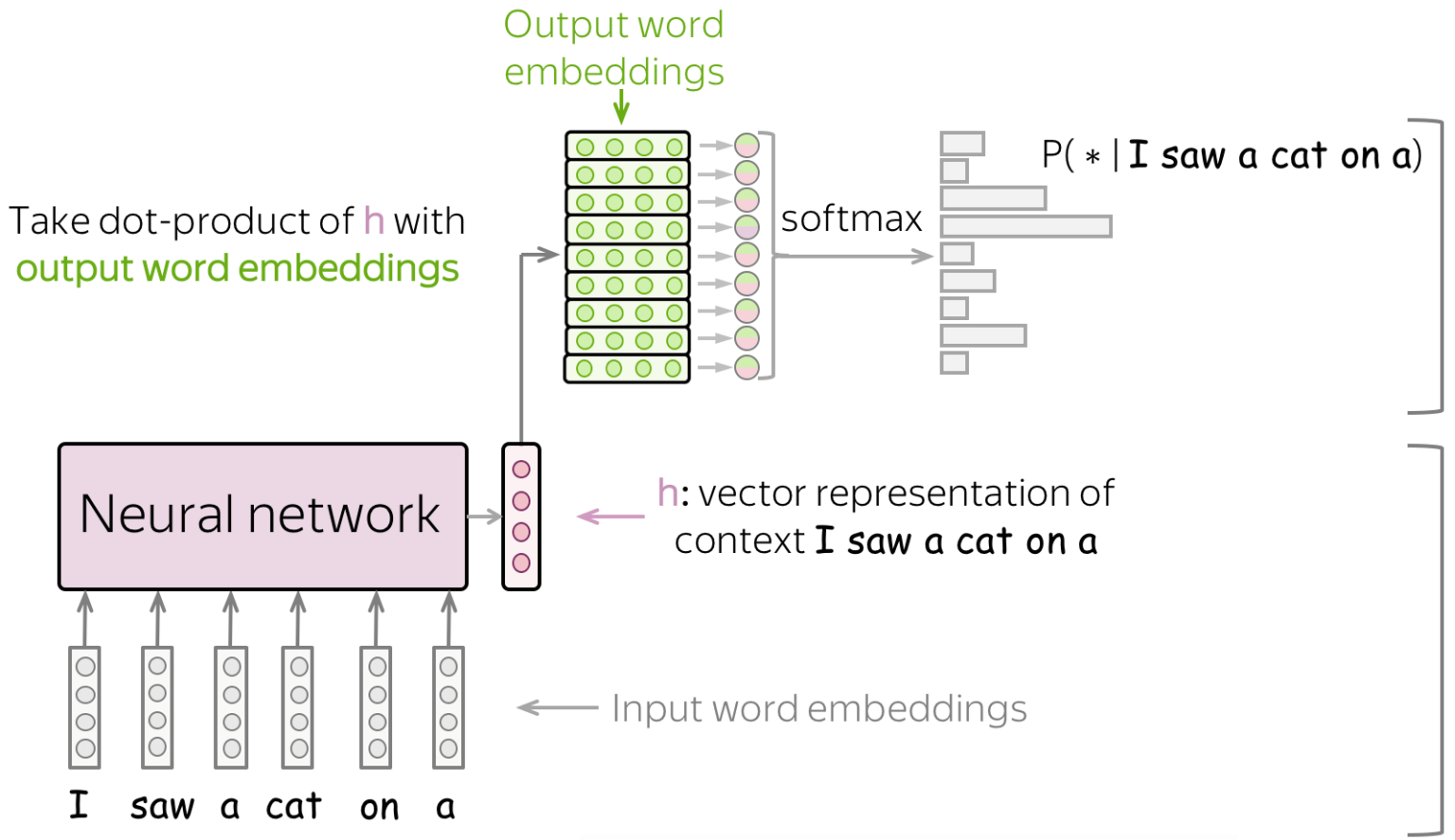
Neural language models has to:

1. *Produce a representation of the prefix*
2. *Generate a probability distribution over the next token*



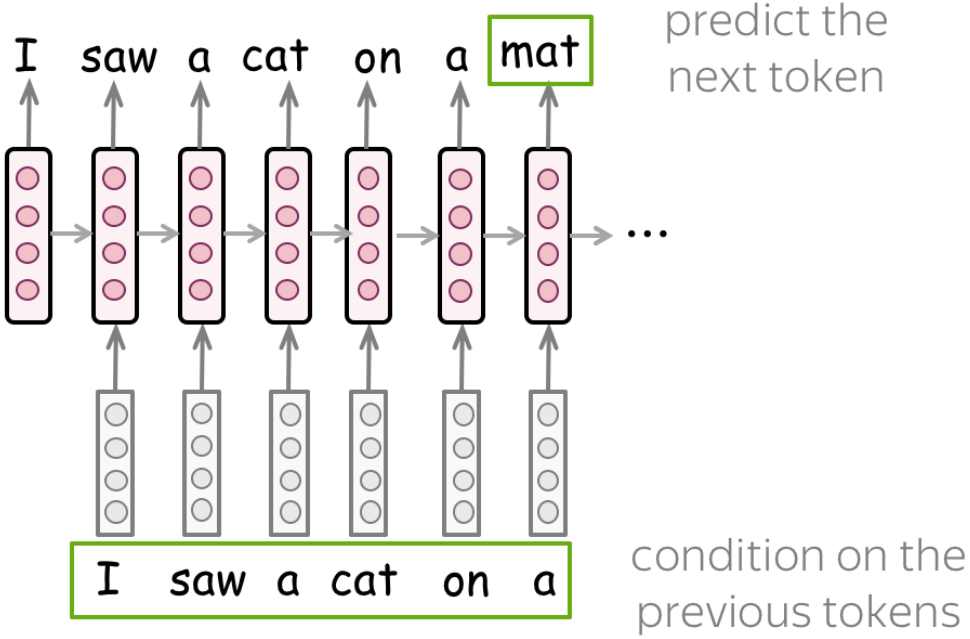
Predicting a word, given a prefix, is just a classification problem!

High-level intuition for a language model

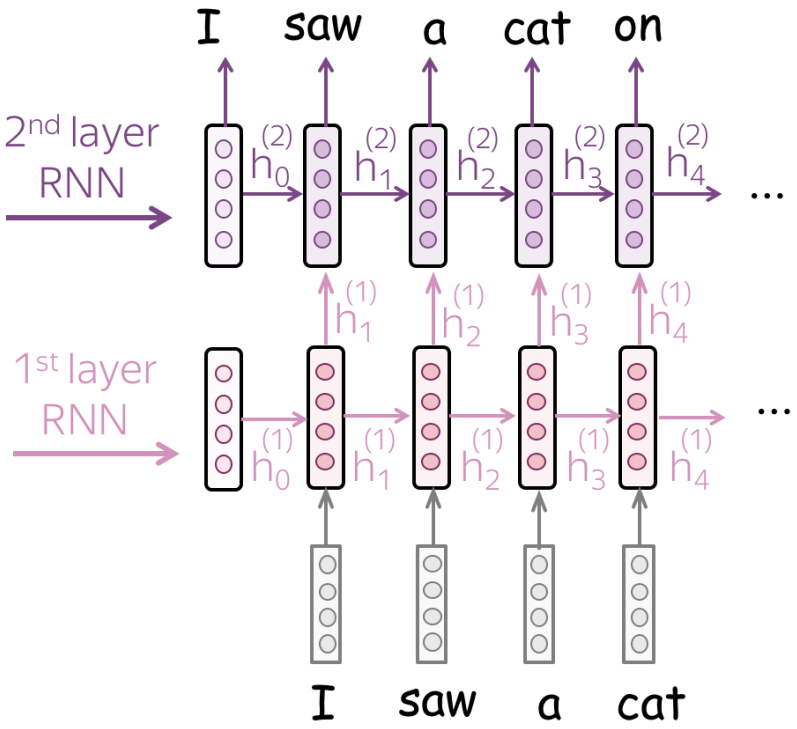


$$p(y_t|y_{<t}) = \frac{exp(h_t^T e_{y_t})}{\sum_{w \in V} exp(h_t^T e_w)}$$

RNN language model



Multi-layer RNN language model



Training the language model

Training is done in a very much the same way as we train a classifier!

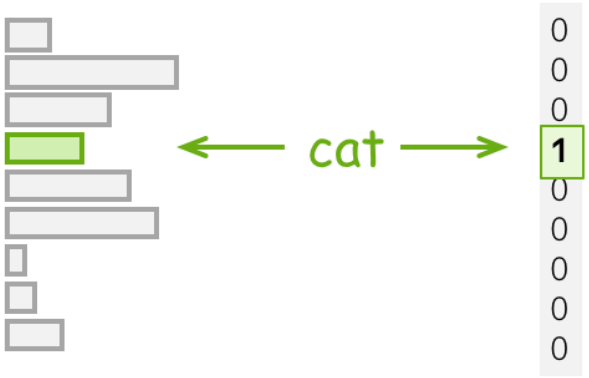
$$Loss = -\log(p(y_t|y_{<t}))$$

we want the model to predict this

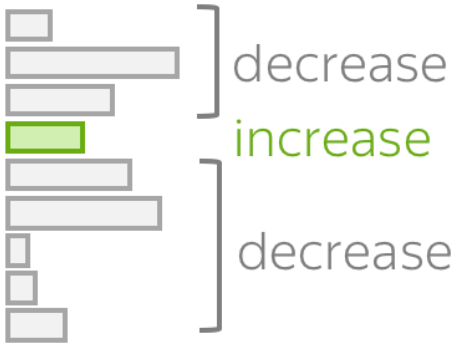
Training example: I saw a cat on a mat <eos>

Model prediction: p(* | I saw a)

Target



Loss = -log (p(cat)) → min



Training for one sentence with RNN LM

(video, not visible in pdf)

RNNs vs Ngram models

Ngram language model

- relies on a short prefix, to get a distribution over next tokens
- explicit independence assumption (can't use context outside of the ngram window)
- smoothing is necessary

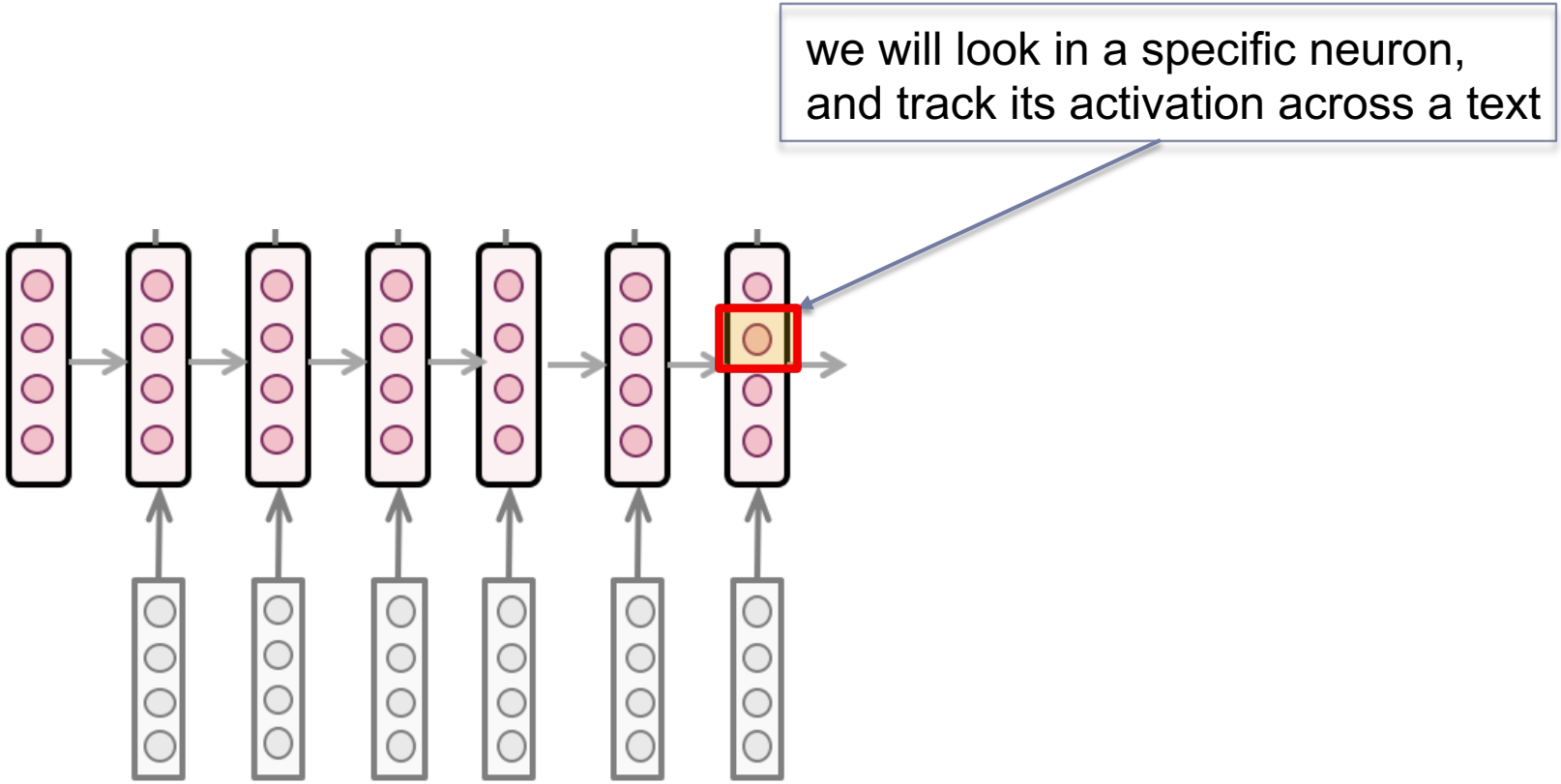
RNN language model

- 'compresses' the past into a state, used to compute the distribution over next tokens
- no independence assumptions; the gradient descent learns to compress the past
- all the information is carried through hidden states (hard to carry it across long distances)

Parallels between RNN state and HMM state learnt in an unsupervised way through EM

Comment: we could also define CNN language models, which would make the explicit independence assumptions but would not require smoothing and has some nice properties

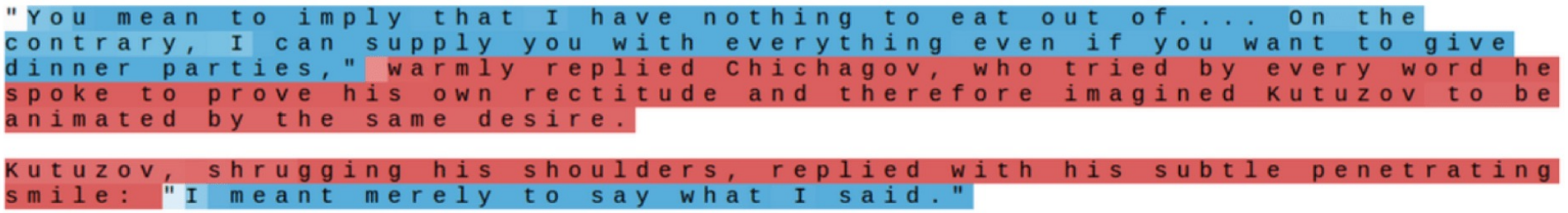
What an RNN does capture in its state?



It is a character-level LM, i.e. models a sequence of characters (rather than words)
Trained on Tolstoy's War and Peace and the source code of Linux Kernel (in C)

Karpathy et el., 2015
https://arxiv.org/abs/1506.02078

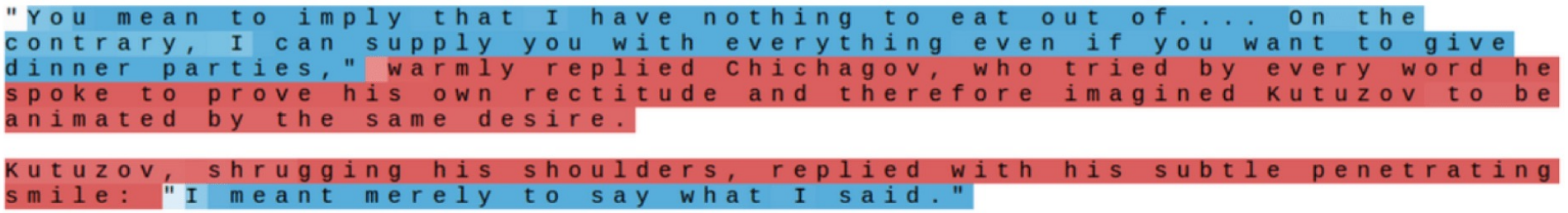
What an RNN does capture in its state?



Any hypothesis what this neuron is doing?

Karpathy et el., 2015
https://arxiv.org/abs/1506.02078

What an RNN does capture in its state?

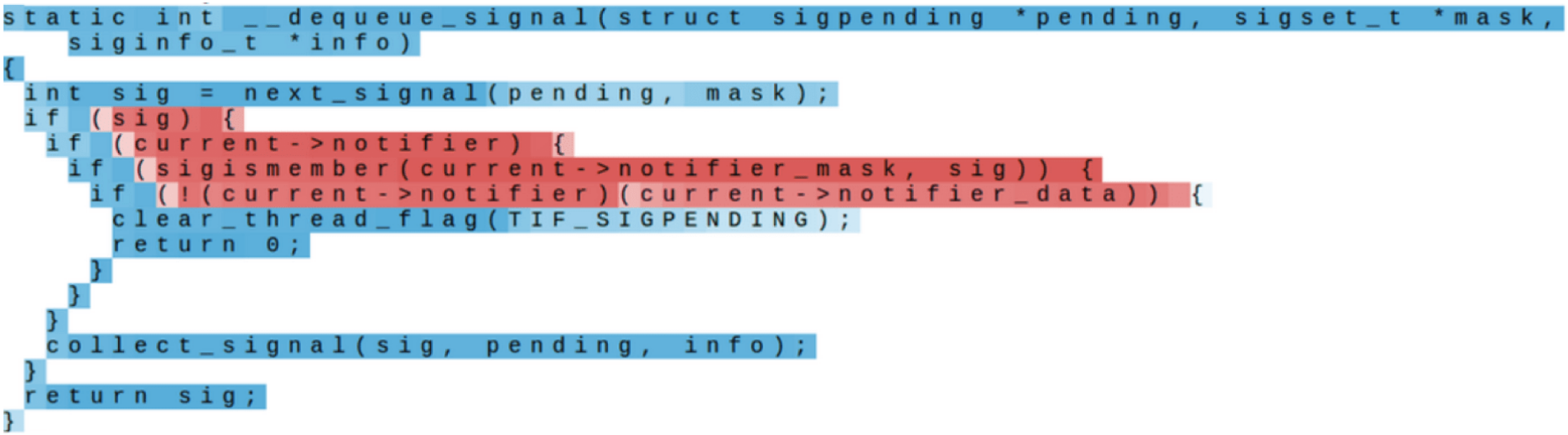


Any hypothesis what this neuron is doing?

It activates within the quotes (“ .. ”)

Karpathy et el., 2015
https://arxiv.org/abs/1506.02078

What an RNN does capture in its state?



```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
        siginfo_t *info)
{
  int sig = next_signal(pending, mask);
  if (sig) {
    if (current->notifier) {
      if (sigismember(current->notifier_mask, sig)) {
        if (!(current->notifier)(current->notifier_data)) {
          clear_thread_flag(TIF_SIGPENDING);
          return 0;
        }
      }
    }
    collect_signal(sig, pending, info);
  }
  return sig;
}
```

Any hypothesis what this neuron is doing?

Karpathy et el., 2015
https://arxiv.org/abs/1506.02078

What an RNN does capture in its state?

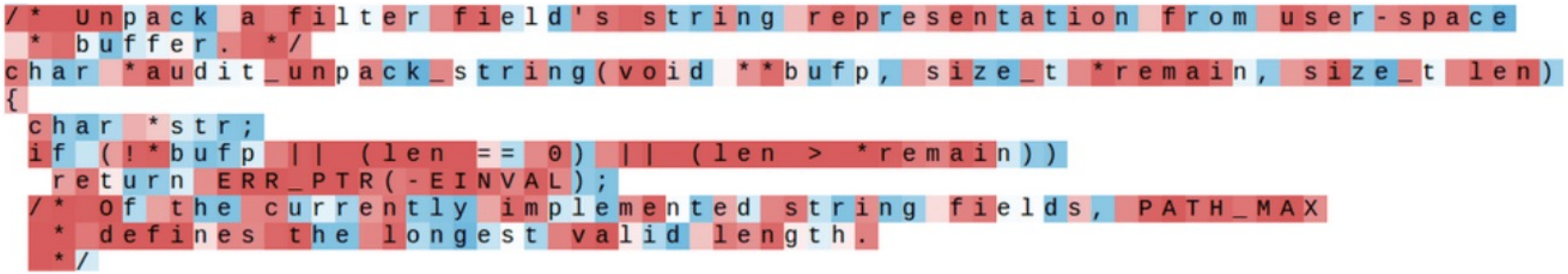
```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
        siginfo_t *info)
{
  int sig = next_signal(pending, mask);
  if (sig) {
    if (current->notifier) {
      if (sigismember(current->notifier_mask, sig)) {
        if (!(current->notifier)(current->notifier_data)) {
          clear_thread_flag(TIF_SIGPENDING);
          return 0;
        }
      }
    }
    collect_signal(sig, pending, info);
  }
  return sig;
}
```

Any hypothesis what this neuron is doing?

Activates within an if statement

Karpathy et el., 2015
https://arxiv.org/abs/1506.02078

What an RNN does capture in its state?

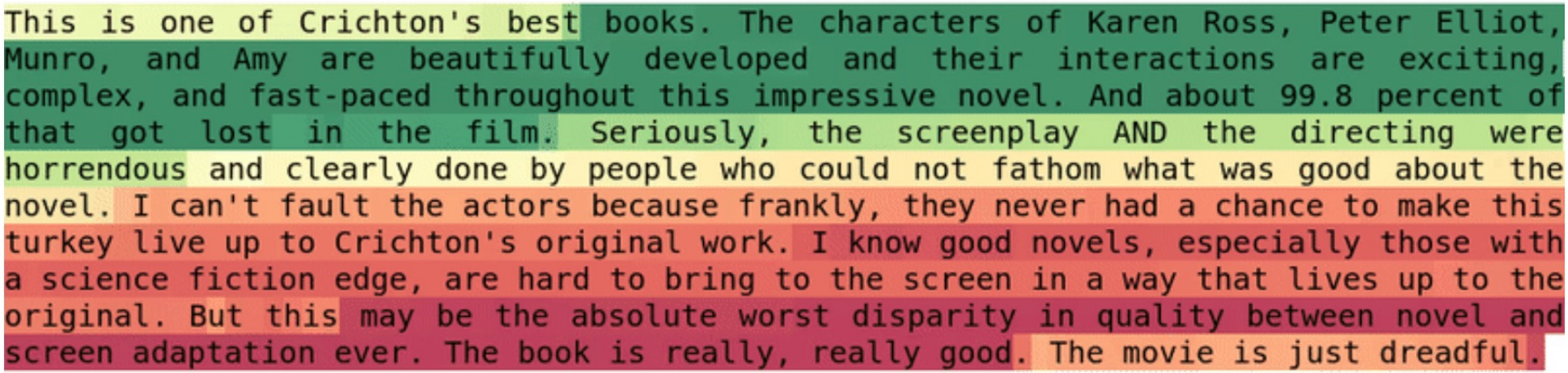


Many neurons are not so easily interpretable

Karpathy et el., 2015
https://arxiv.org/abs/1506.02078

Sentiment neuron

This is from a much bigger LSTM model trained by OpenAI on Amazon reviews



They could also switch the sentiment at generation time to change the sentiment of an utterance (call *interventions*)

https://openai.com/research/unsupervised-sentiment-neuron

Do RNNs learn syntax?

Remember, 10 lectures ago, we observed that ngrams are not able to capture syntactic agreement

Sam/Dogs sleeps/sleep soundly
Sam, who is my cousin, sleeps soundly
Dogs often stay at my house and sleep soundly
Sam, the man with red hair who is my cousin, sleeps soundly

We argued that syntax is needed. But then using PCFG in generation is virtually impossible, can neural networks accomplish this?

The roses in the vase by the door __?__ Competing answers: is, are

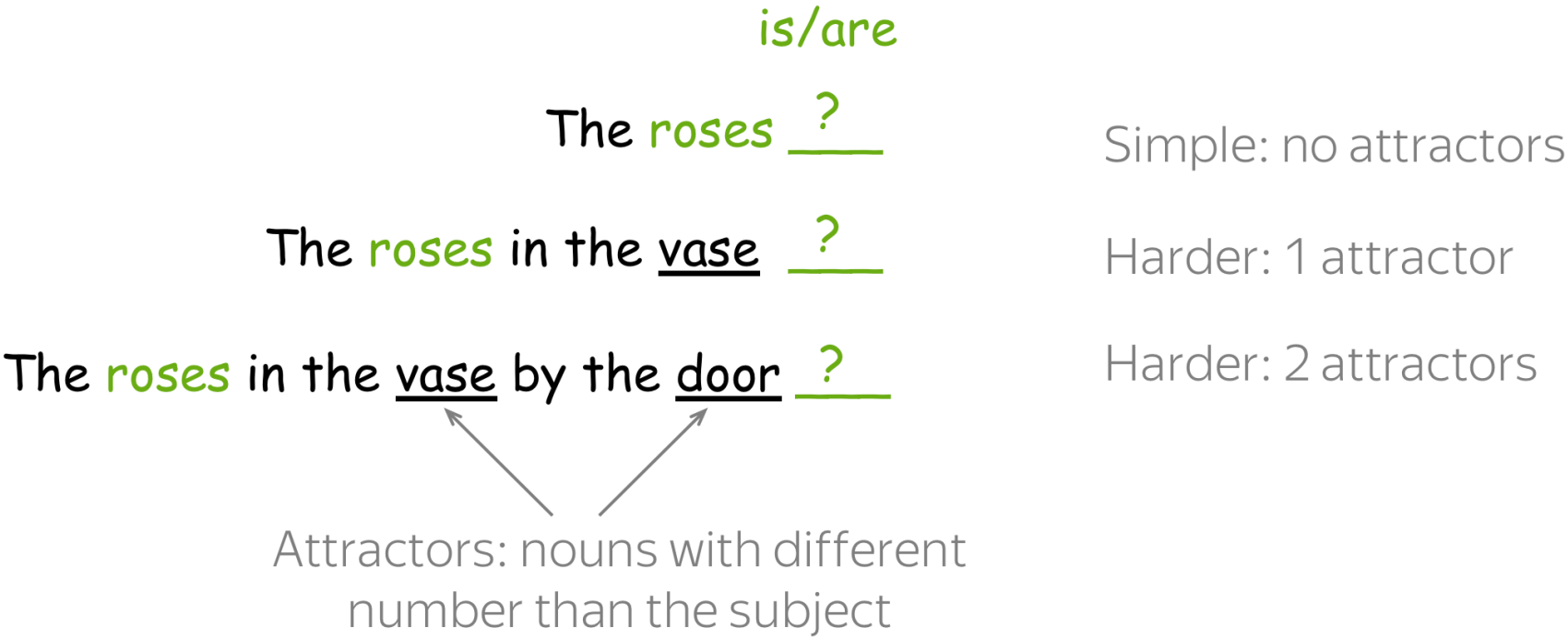
P(The roses in the vase by the door are)

P(The roses in the vase by the door is)

Is the correct answer ranked higher?

P(...are) > P(...is)?

Contrastive evaluation



Short summary:

- need to be careful to prevent the model from relying on non-syntactic 'shortcuts'
- LSTMs models trained for language modeling were not as strong in that evaluation (but more powerful models will be)

Linzen, Dupoux and Goldberg, 2016
https://aclanthology.org/Q16-1037.pdf

Summary

- Recurrent neural networks can capture long-distance dependences in text
- A neural language model is a multiclass classifier
- Some of the neurons in RNNs are interpretable