# Sentiment Analysis and Introduction to RNNs

Partially based on slides by Jurafsky and Martin

Speech and Language Processing, 3rd Edition

# Sentiment Analysis

*Example #1: Movie Reviews*

- Unbelievably disappointing

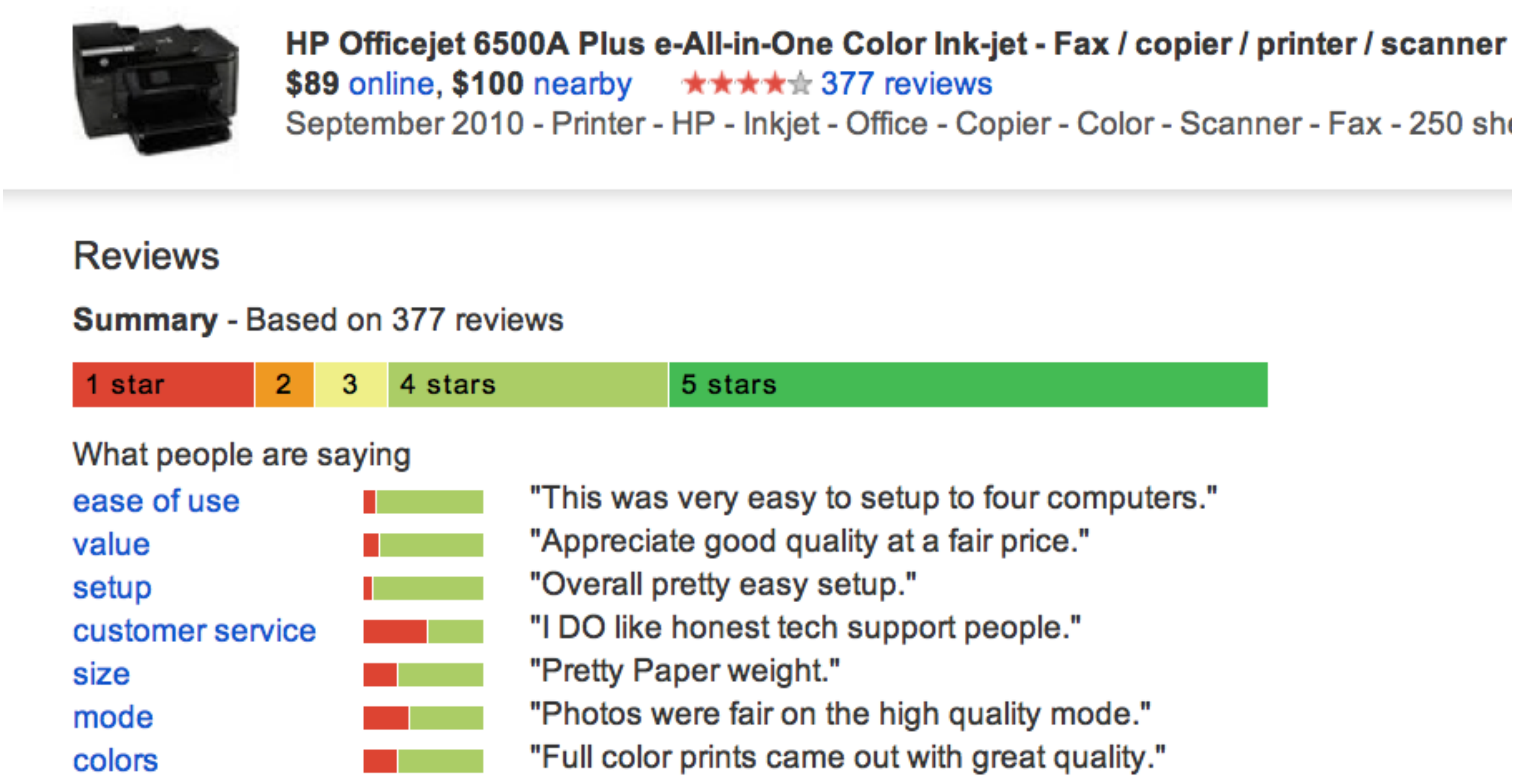
- Full of zany characters and richly applied satire, and some great plot twists

- This is the greatest screwball comedy ever filmed

- It was pathetic. The worst part about it was the boxing scenes.

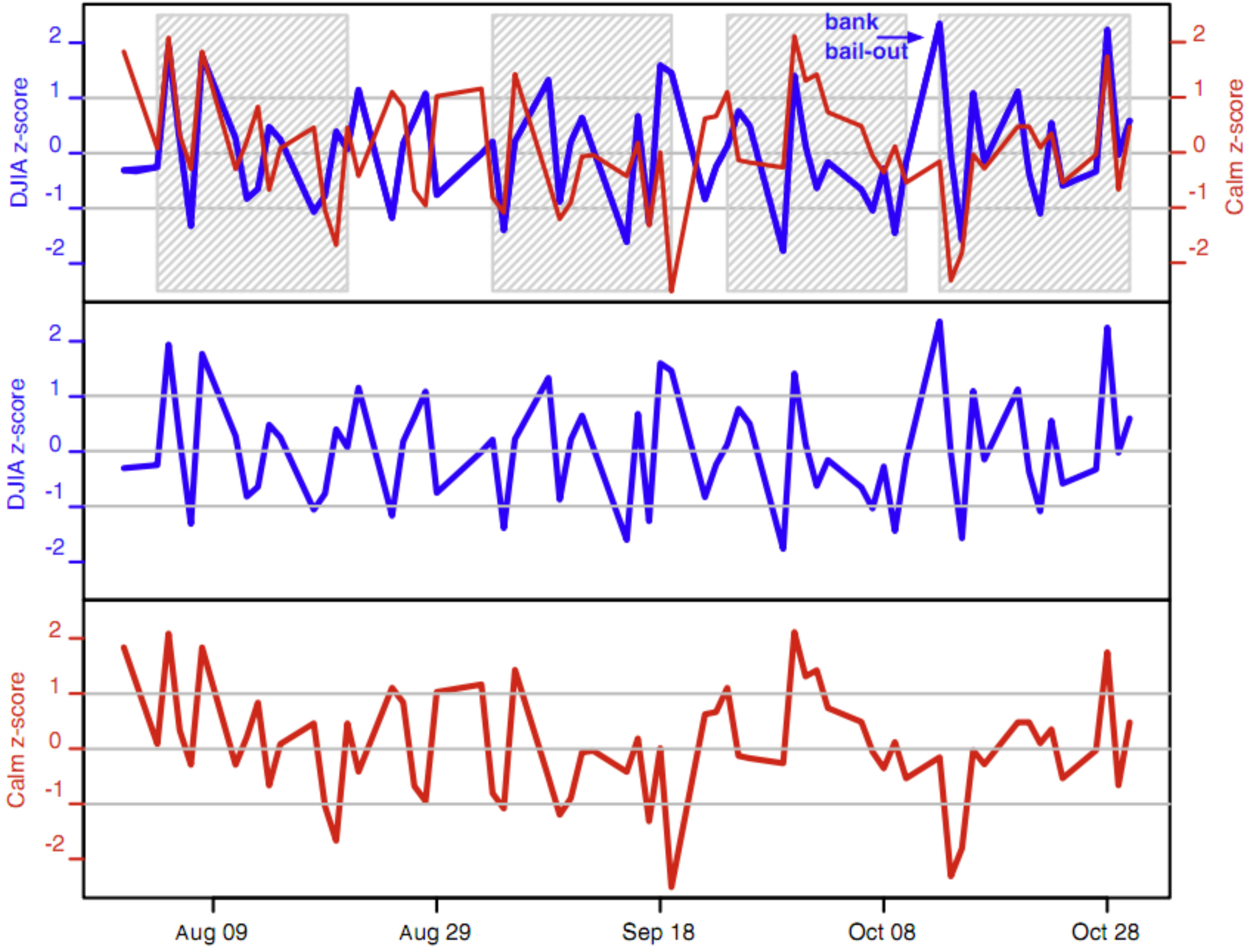
# Sentiment Analysis

*Example #2: Product Reviews*



- A Sentiment Analysis system called CALM applied to Twitter predicts the Dow Jones Industrial Average (DJIA) 3 days later
- Such algorithms are already used by hedge funds

Johan Bollen, Huina Mao, Xiaojun Zeng. 2011. [Twitter mood predicts the stock market,]() Journal of Computational Science 2:1, 1-8. 10.1016/j.jocs.2010.12.007.



# Scherer Typology of Affective States

- **Emotion**: brief organically synchronized … evaluation of a major event
  - *angry, sad, joyful, fearful, ashamed, proud, elated*
- **Mood**: diffuse non-caused low-intensity long-duration change in subjective feeling
  - *cheerful, gloomy, irritable, listless, depressed, buoyant*
- **Interpersonal stances**: affective stance toward another person in a specific interaction
  - *friendly, flirtatious, distant, cold, warm, supportive, contemptuous*
- **Attitudes**: enduring, affectively colored beliefs, dispositions towards objects or persons
  - *liking, loving, hating, valuing, desiring*
- **Personality traits**: stable personality dispositions and typical behavior tendencies
  - *nervous, anxious, reckless, morose, hostile, jealous*

# Sentiment Analysis: Definition

- Sentiment analysis is the detection of **attitudes**

  “enduring, affectively colored beliefs, dispositions towards objects or persons”

  1. **Holder (source)** of attitude
  2. **Target (aspect)** of attitude
  3. **Type** of attitude
     - From a set of types: *like, love, hate, value, desire,* etc.
     - Or (more commonly) simple weighted **polarity**: *positive, negative, neutral,* together with *strength*
  4. **Text** containing the attitude
     - Sentence or entire document

# Sentiment Analysis

- Simplest task:
  - Is the attitude of this text positive or negative?
- More complex:
  - Rank the attitude of this text from 1 to 5
- Advanced:
  - Detect the target, source, or complex attitude types

- Sometimes not only complete sentences, but also their sub-parts receive a sentiment value
  - [The pizza was overall bad], but the toppings were fun! → overall negative

# Sentiment Classification in Movie Reviews

- Is an IMDB movie review positive or negative?

✓

when _star wars_ came out some twenty years ago , the image of traveling throughout the stars has become a commonplace image . [...]

when han solo goes light speed , the stars change to bright lines , going towards the viewer in lines that converge at an invisible point .

cool .

✗

" snake eyes " is the most aggravating kind of movie : the kind that shows so much potential then becomes unbelievably disappointing .

it's not just because this is a brian depalma film , and since he's a great director and one who's films are always greeted with at least some fanfare .

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. EMNLP-2002, 79—86.
Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. ACL, 271-278

# A Simple Classifier

- Log-linear or Naïve Bayes classifier
- Features:
  - Tokenized words
  - Possibly mark-up (e.g., hashtags in Twitter, headers in HTMLs)
- Features are often binary
  - Indicating whether a word appeared or did not appear in the document (bag of words)
  - Often works better for text classification than word frequency
    - The effect of frequency is non-linear

# Error Analysis: What makes reviews hard to classify?

- **Sarcasm:**
  - Perfume review in *Perfumes: the Guide*:
    - "If you are reading this because it is your darling fragrance, please wear it at home exclusively, and tape the windows shut."
  - On *Automobile Steering Wheel Attachable Work Surface*:
    - "You wouldn't believe how much more interesting my commute is now that I have something to do other than just stare out the window! I'm using it right now to post this review and I never"

- **Thwarted Expectations and Ordering Effects:**
  - "This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it **can't hold up**."

# Negation in Sentiment Analysis

- One practice is to add NOT_ to every word between negation and following punctuation:

```
didn't like this movie , but I

didn't NOT_like NOT_this NOT_movie but I
```

# Negation in Sentiment Analysis

- This is a very crude solution:
  - Explicit negation is only one way to reverse polarity
    - “He **failed** to convey the importance of his message”
  - Negation scope:
    - “I did**n’t** like the exposition in this otherwise excellent film”
  - Logical structure can be more complex
    - “I do**n’t** think anyone could have done a better job”

- More recent approaches take the context (surrounding words) into account

# Sentiment Analysis as Sequence Labeling

- This construal of sentiment analysis attempts to capture the meaning of a word <u>in context</u> by encoding (parts of) the sentence as features
- Recently: using Recurrent Neural Networks (RNNs)

- Recall the underlying assumption in Markov (n-gram) models:
  - It's enough to know the last *n* tokens you've encountered to know what's next
  - Alternative view: the probability of a sequence is the product of the probabilities of its *n-grams*

# Sentiment Analysis as Sequence Labeling

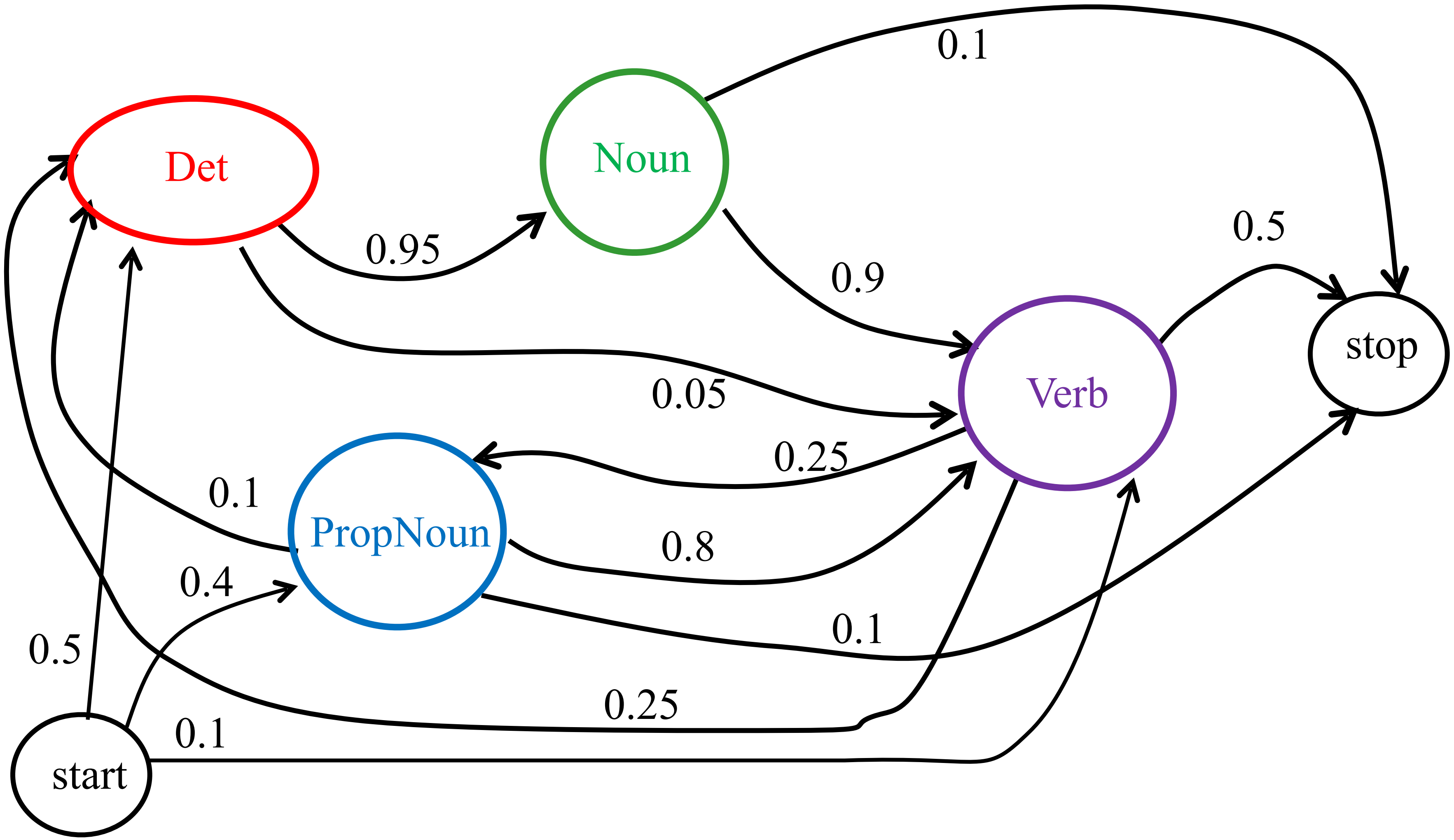
- Consider the example:
  - *How can you not see this movie?*
  - *You should not see this movie*

  - Great idea, you have a real product development team!
  - Great idea, now try again with a real product development team!

- How well will a bi-gram model work?
  - Similar unigrams and bigrams → similar prediction

- The problem with Markov Models: need to maintain a **state** to capture distant influences
  - The size of the space increases exponentially with the order of the model

# Recall: Markov Models are Finite State Automata (FSAs) with transition probabilities



# Recurrent Neural Networks

- Motivation:
  - Neural network model, but with a state
  - How can we borrow ideas from FSAs?
- RNNs are FSAs …
  - With a twist
  - No longer finite in the same sense
  - The state is an embedding of the history in a continuous space
  - The embedding function of the history to a vector is learned as well

# Recurrent Neural Networks

- Map from dense sequence to dense representation
  - Maps a sequence of vector inputs to a sequence of vector states

$$x_1, \ldots, x_n \rightarrow s_1, \ldots, s_n$$

  - A (parametrized) transition function *R* does the mapping:

$$s_i = R(s_{i-1}, x_i)$$

  - *R* is parametrized and parameters are shared across steps

$$s_4 = R(s_3, x_4) = R(R(s_2, x_3), x_4) = R(R(R(s_1, x_2), x_3), x_4)$$

# Recurrent Neural Networks

- An output function *O* maps states to (vector) outputs, which are often viewed as a distribution over the possible labels

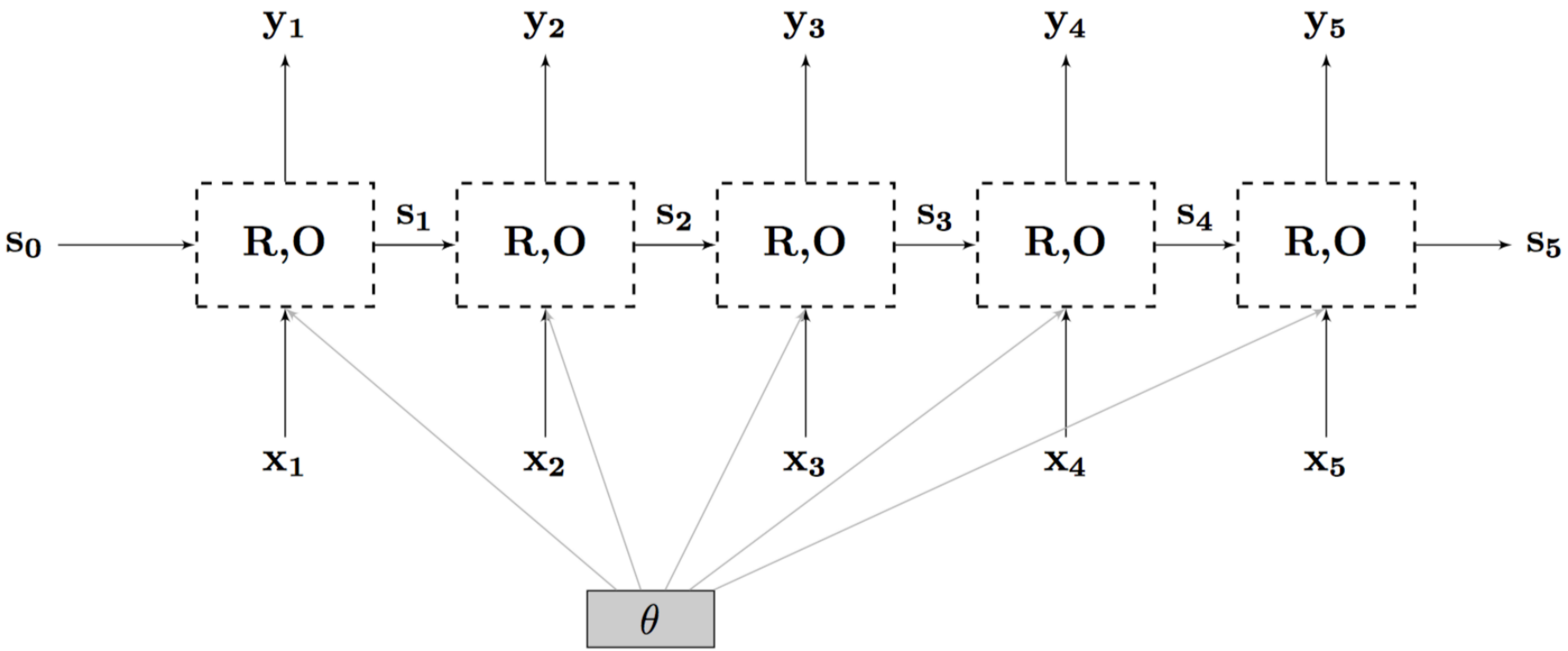
- Example:

$$R_{Elman}(s, x) = tanh(W[x, s] + b)$$

$$O(s_i) = softmax(W' s_i + b')$$

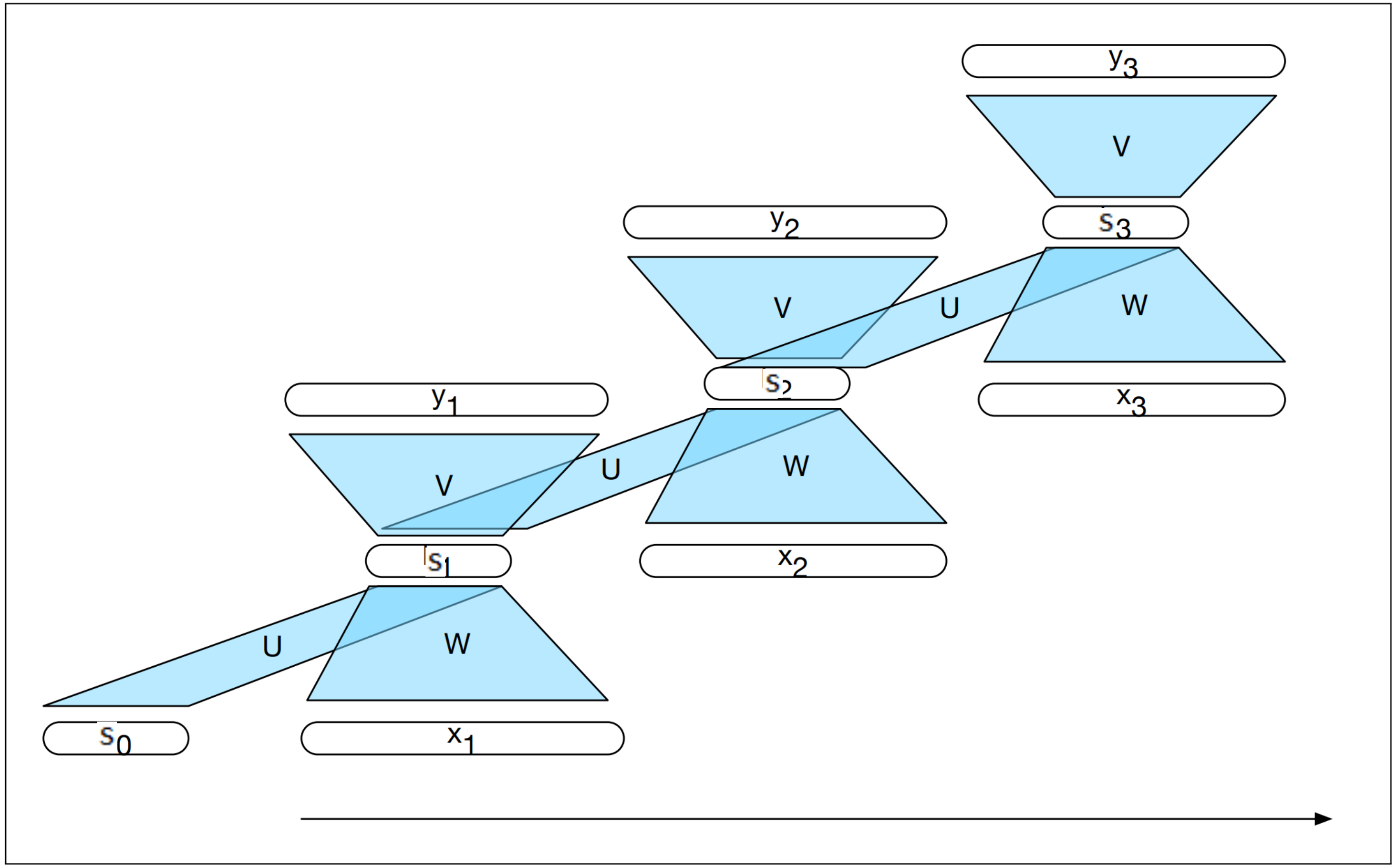
- Conceptually:
  - *R* is the function embedding the history in a vector
  - *O* is the function relating the embedded history and the output
  - They are learned <u>jointly</u>

# RNNs: Graphical Representation



# Recurrent Neural Language Models

- An “unrolled” view of an RNN
- The network stays the same at each time stamp. What’s different are the inputs and previous hidden states



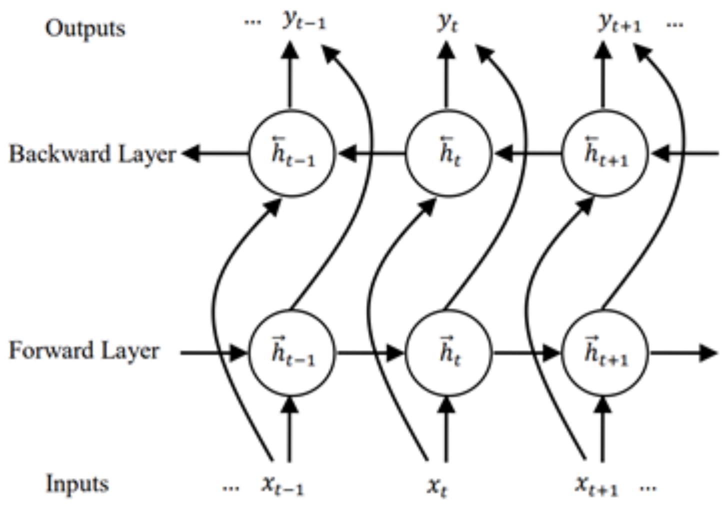
# Back to Sentiment Analysis

- When using RNNs for sentence classification (such as sentiment analysis), it is often practical to use Bi-RNNs
- Bi-RNNs:
  - 2 RNNs, one going back to forth and the other forth to back
  - Output function is a function of both states
- This allows the states associated with each word to encode relevant information from the words following them and preceding them

$$\overrightarrow{h}_t = (W_{x\overrightarrow{h}} x_t + W_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}) \quad (9)$$

$$\overleftarrow{h}_t = (W_{x\overleftarrow{h}} x_t + W_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}) \quad (10)$$

$$y_t = W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y \quad (11)$$




"Contextual Bidirectional Long Short-Term Memory Recurrent Neural Network Language Models: A Generative Approach to Sentiment Analysis", Mousa and Shuller, EACL 2017

# Back to Sentiment Analysis

- One simple way to do sentiment analysis (or other sentence classification) with Bi-RNNs is to average the output sequence:

$$y = \frac{1}{N}\sum_i y_i$$

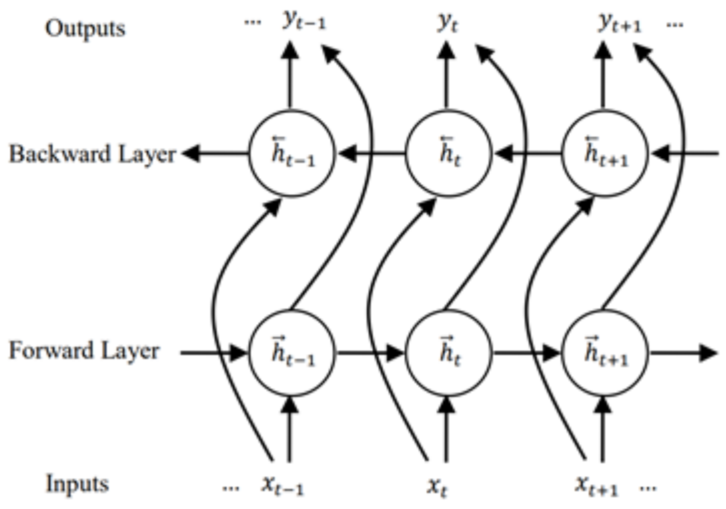
- Now train a binary (log-linear) classifier for predicting the sentiment:

$$P(+|y) = \frac{1}{1+e^{-(w^t \cdot y + b)}} = sigmoid(w^t \cdot y + b)$$

$$\overrightarrow{h}_t = (W_{x\overrightarrow{h}} x_t + W_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}) \tag{9}$$

$$\overleftarrow{h}_t = (W_{x\overleftarrow{h}} x_t + W_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}) \tag{10}$$

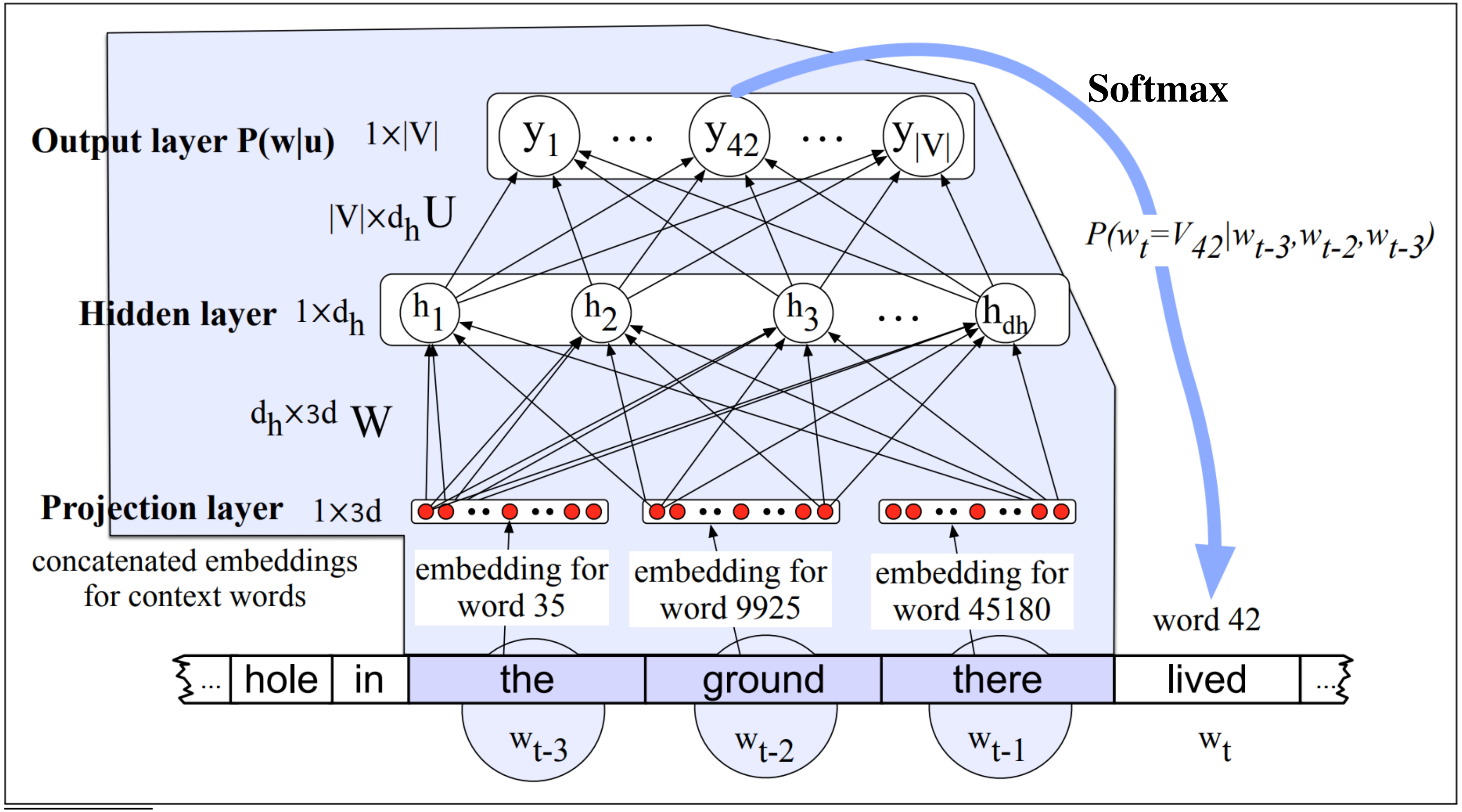
$$y_t = W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y \tag{11}$$



# Context in RNN models for Sentiment Analysis

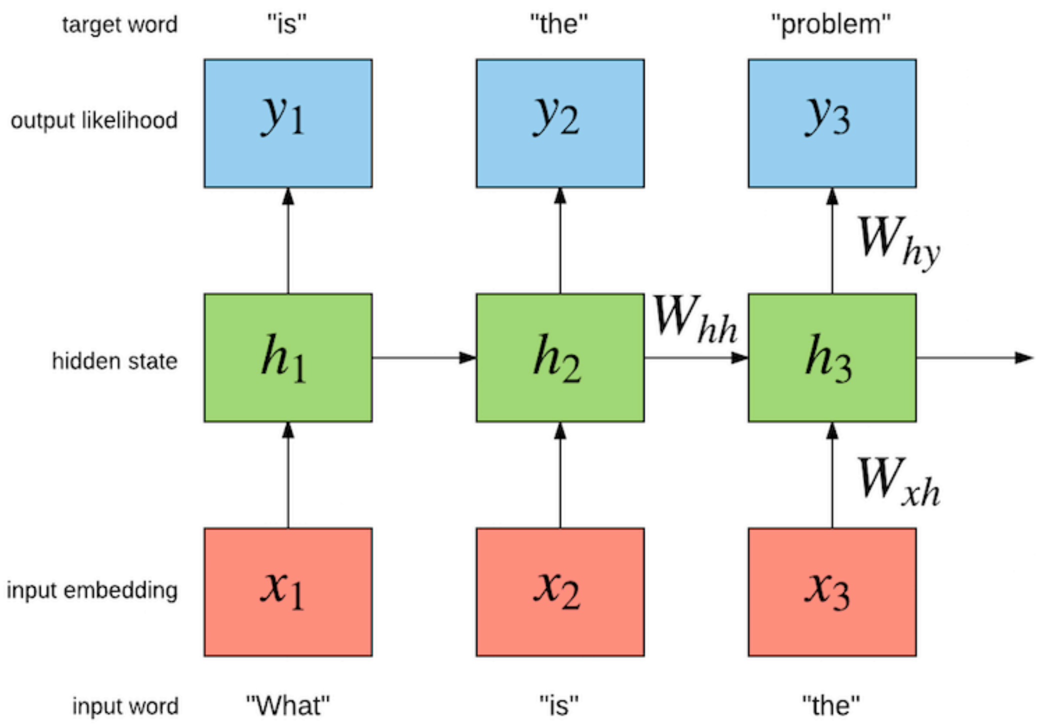
- RNN models are able to take the context (both preceding and following) into account, as well as the linear order between the words
  - bag-of-words models cannot
- They indeed show better performance in tasks such as sentiment analysis
- However, the state sequence is not easy to interpret
  - Much heavier computationally, especially training
- Further investigation is needed to establish what contextual and semantic aspects of sentences are captured using these techniques

# Feed Forward Neural Language Models



# Recurrent Neural Language Models

- Language models based on RNNs have shown much power in recent years, consistently surpassing n-gram models
- The basic architecture is that of a sequence recurrent neural net (RNN)
- Just like in Feed Forward LMs:
    - Input words are converted embeddings
    - The output is passed through a softmax layer, which defines the probability of the next word
- The difference is that the hidden state is passed as input to the new layer




http://www.fit.vutbr.cz/~imikolov/rnnlm/thesis.pdf - Mikolov (2011)

# Reminder: Loss Functions

- Training is done by defining a loss function (i.e. a function that determines how "bad" a classification is relative to the gold standard)
- The network then attempts to minimize the empirical risk. For a sample $S=\{x_1,\dots,x_m\}$ with gold standard labels $\{y_1,\dots,y_m\}$, the empirical risk is:

$$L_S(\theta) = \frac{1}{m}\sum_{i=1}^{m} \ell(\theta; (x_i, y_i))$$

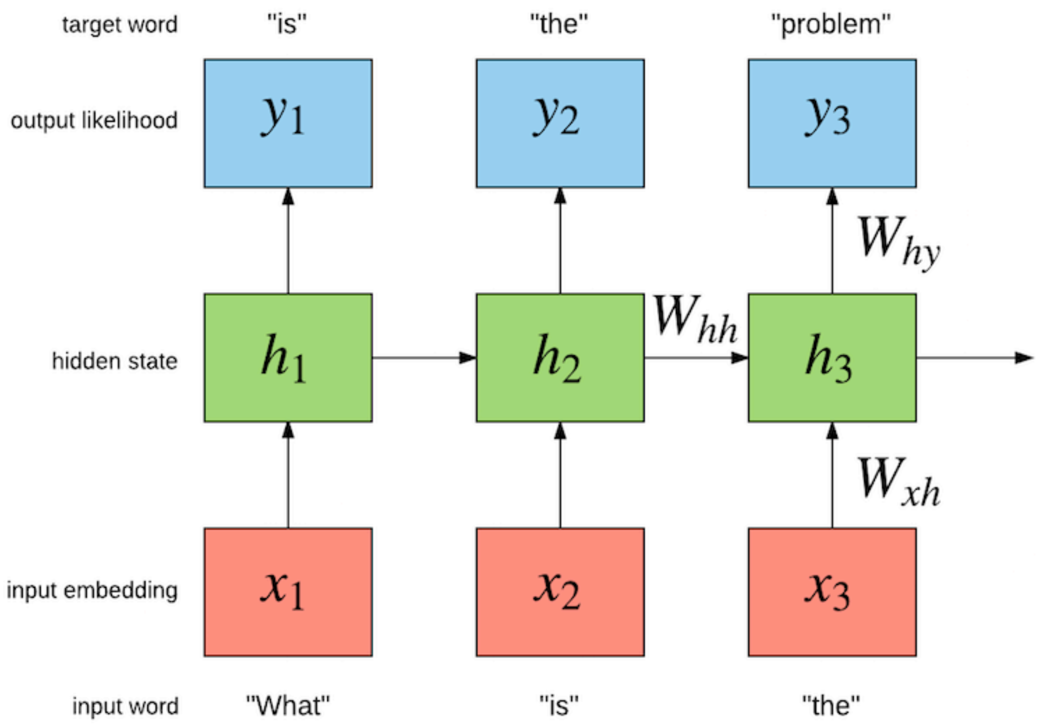
# Recurrent Neural Language Models

- Neural language models are generally trained to maximize the sum of the predicted log-probabilities of the correct words

  → the loss function is the negative predicted log-probabilities of the correct words

- In the example to the right, the loss will be

$$-log(soft\_max(y_1)(is))-$$
$$log(soft\_max(y_2)(the))-$$
$$log(soft\_max(y_3)(problem))$$

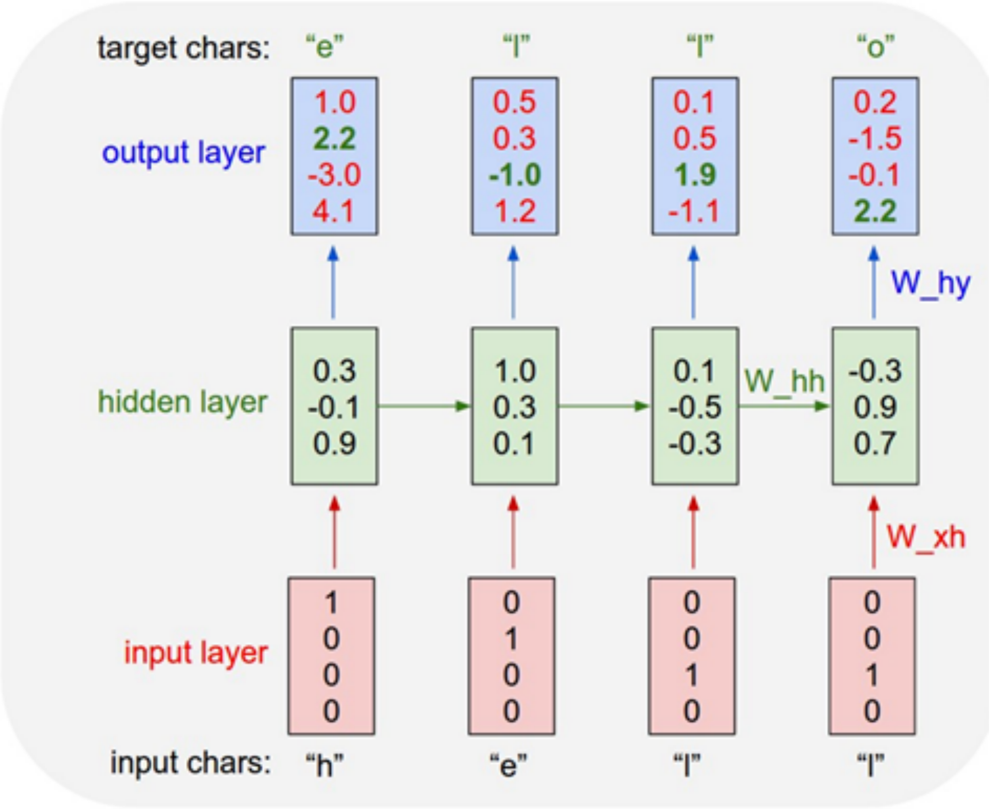



http://www.fit.vutbr.cz/~imikolov/rnnlm/thesis.pdf - Mikolov (2011)

# Character-level Language Models

- Vocabulary: characters instead of words
- Advantage:
  - Small vocabulary → compact model
  - Can generalize over morphologically similar words
- However:
  - Need to learn how to spell
  - Longer range dependencies between tokens

# Character-level Language Models




[during training – green = value to increase]

http://karpathy.github.io/2015/05/21/rnn-effectiveness/