(Recurrent) Neural Networks

and Applications

Data Mining & Analytics

Prof. Zach Pardos

INFO254/154: Spring ’19

Class notes

- RNN Lab Published (10 extra credit points) using an NLP dataset
- May 3rd - last day for late lab submission (re-submission)

Think of datasets, scenarios, domains, problems involving time-series

(where the future is a function of the present and the past)

Keras simple NN implementation

Simple 3 layer NN

```
from keras.layers import Input, Dense
from keras.models import Model

# This returns a tensor
inputs = Input(shape=(784,))

# a layer instance is callable on a tensor, and returns a tensor
x = Dense(64, activation='relu')(inputs)
x = Dense(64, activation='relu')(x)
predictions = Dense(10, activation='softmax')(x)

# This creates a model that includes
# the Input layer and three Dense layers
model = Model(inputs=inputs, outputs=predictions)
model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
model.fit(data, labels)  # starts training
```

Keras RNN/LSTM implementation

RNN / LSTM example

```
# as the first layer in a Sequential model
model = Sequential()
model.add(LSTM(32, input_shape=(10, 64)))
# now model.output_shape == (None, 32)
# note: `None` is the batch dimension.

# for subsequent layers, no need to specify the input size:
model.add(LSTM(16))

# to stack recurrent layers, you must use return_sequences=True
# on any recurrent layer that feeds into another recurrent layer.
# note that you only need to specify the input size on the first layer.
model = Sequential()
model.add(LSTM(64, input_dim=64, input_length=10, return_sequences=True))
model.add(LSTM(32, return_sequences=True))
model.add(LSTM(10))
```

Extra code examples

Feed-forward neural network

<u>Single-layer</u> Perceptron

Vocabulary size = 3
Nodes in output layer = 2

$X =$

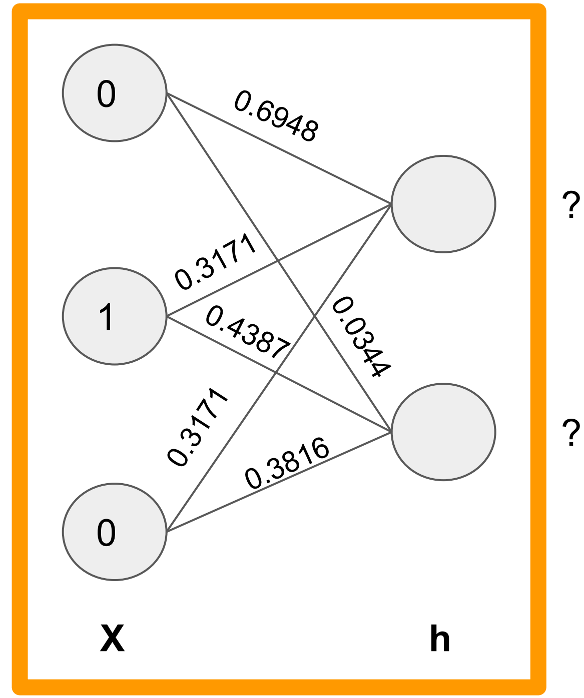
0	1	0

$W_{xh} =$

0.6948	0.0344
0.3171	0.4387
0.9502	0.3816

$h = W'_{xh}X' =$

?	?



No bias, no squashing (activation) function

Feed-forward neural network

Single-layer Perceptron

Vocabulary size = 3
Nodes in output layer = 2

$X =$

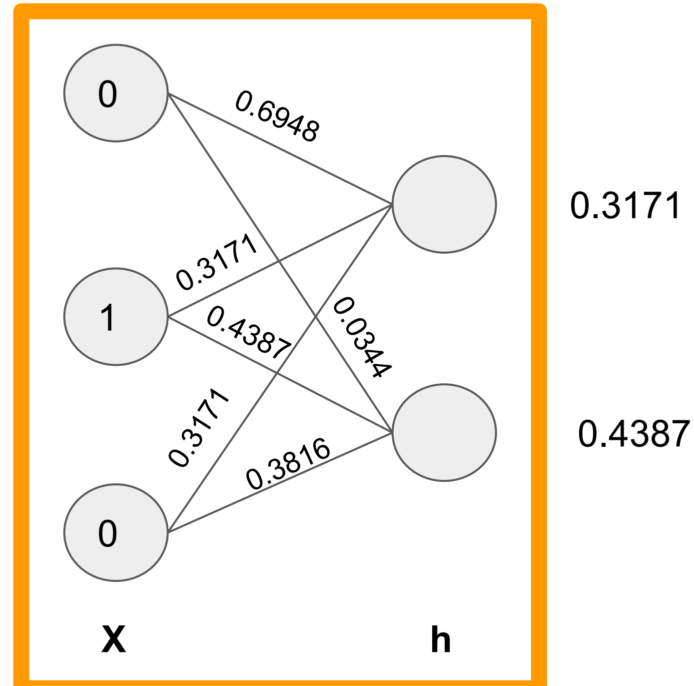
0	1	0

$W_{xh} =$

0.6948	0.0344
0.3171	0.4387
0.9502	0.3816

$h =$

0.3171	0.4387



No bias, no squashing (activation) function

Feed-forward neural network

Single-layer Perceptron

Vocabulary size = 3
Nodes in **hidden** layer = 2

$X =$

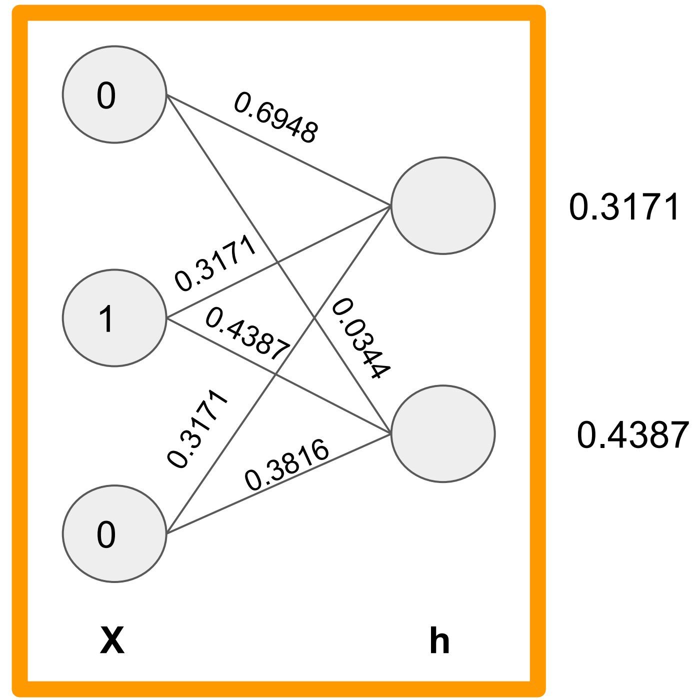
0	1	0

$W_{xh} =$

0.6948	0.0344
0.3171	0.4387
0.9502	0.3816

$h =$

0.3171	0.4387



Feed-forward neural network

<u>Multilayer</u> Perceptron

Input size = 3
Output size = 2

$X =$

0	1	0

$W_{xh} =$

0.6948	0.0344
0.3171	0.4387
0.9502	0.3816

$h =$

0.3171	0.4387

$W_{ho} =$

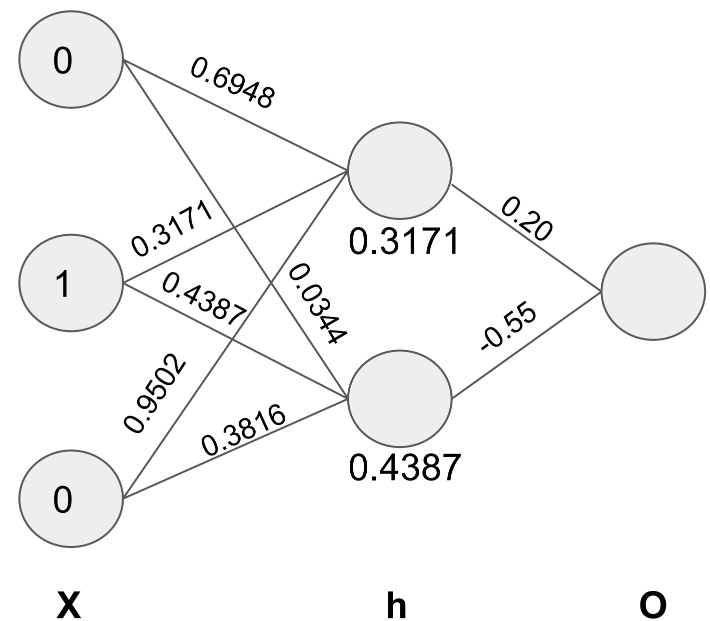
0.20
-0.55

$O = h*W_{ho} = -0.1779$

sigmoid(O) =

0.4556

No bias



Feed-forward neural network

Multilayer Perceptron / Skip-gram

Input size = 3
Output size = 3

$X =$

0	1	0

$W_{xh} =$

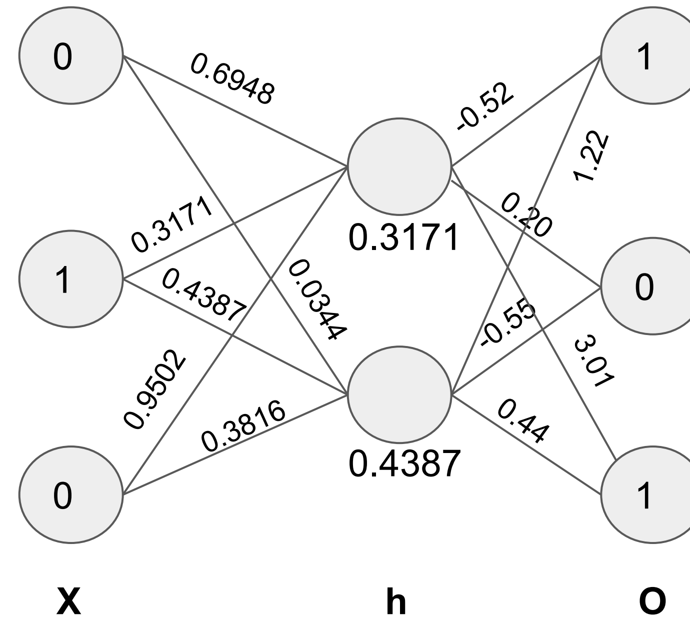
0.6948	0.0344
0.3171	0.4387
0.9502	0.3816

$h =$

0.3171	0.4387

$W_{ho} =$

-0.52	0.20	3.01
1.22	-0.55	0.44



Feed-forward neural network

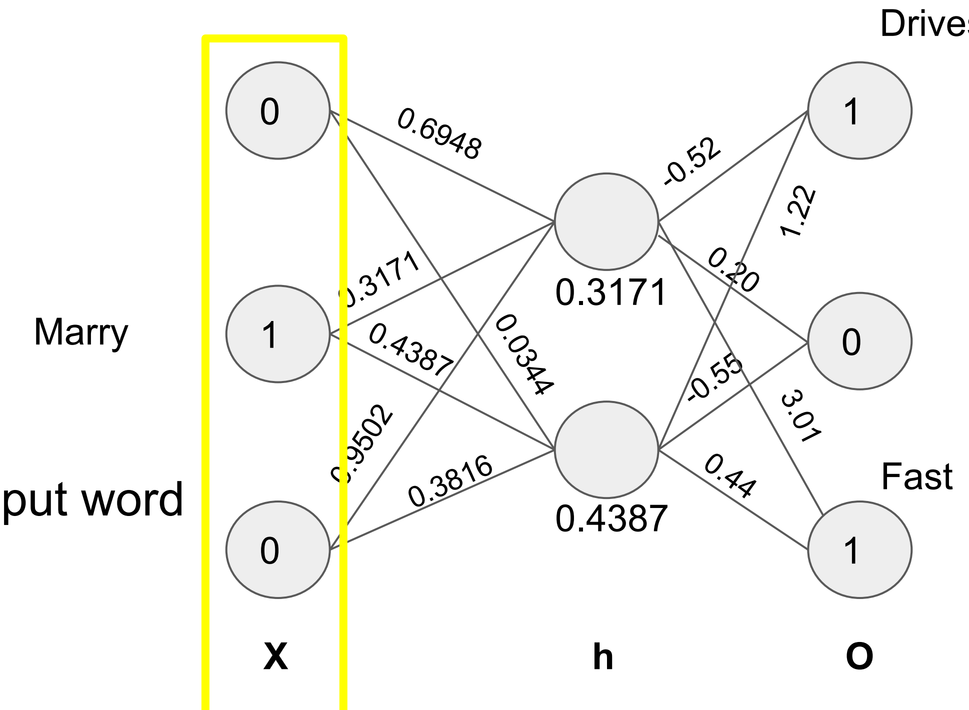
Multilayer Perceptron / <u>Skip-gram</u>

Input size = 3
Output size = 3

Word input one-hot

Consider a sentence:
Marry drives fast

Skip-gram predicts the context of the input word

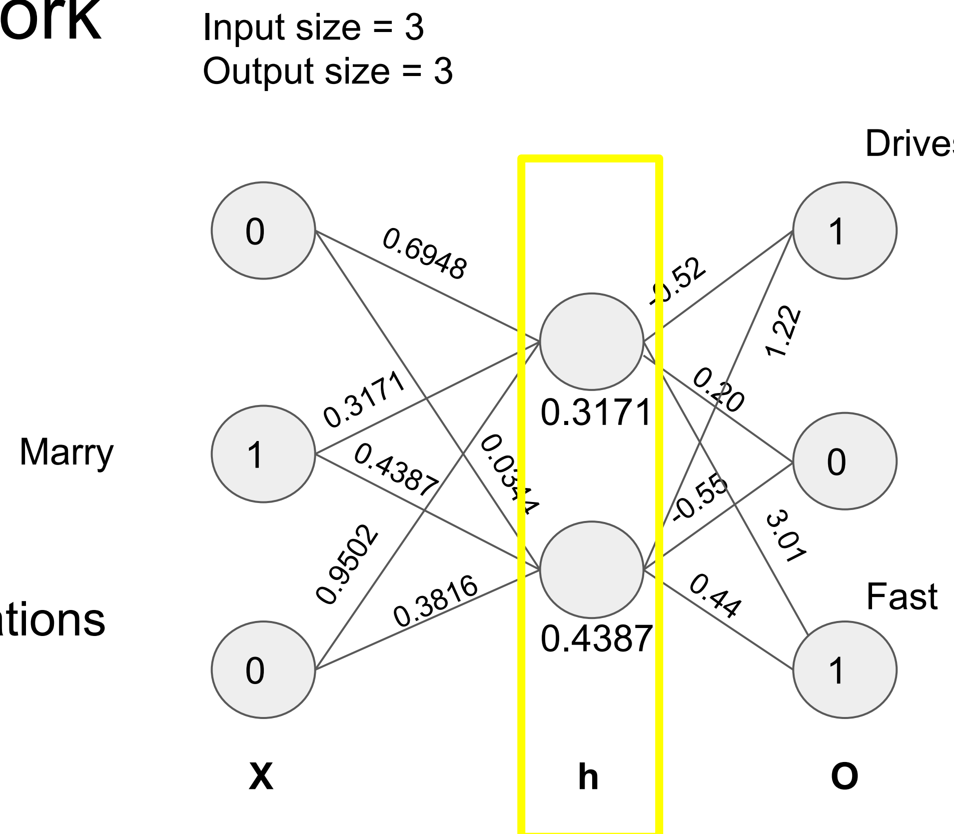


Feed-forward neural network

Multilayer Perceptron / Skip-gram

Continuous representation of word (embedding)

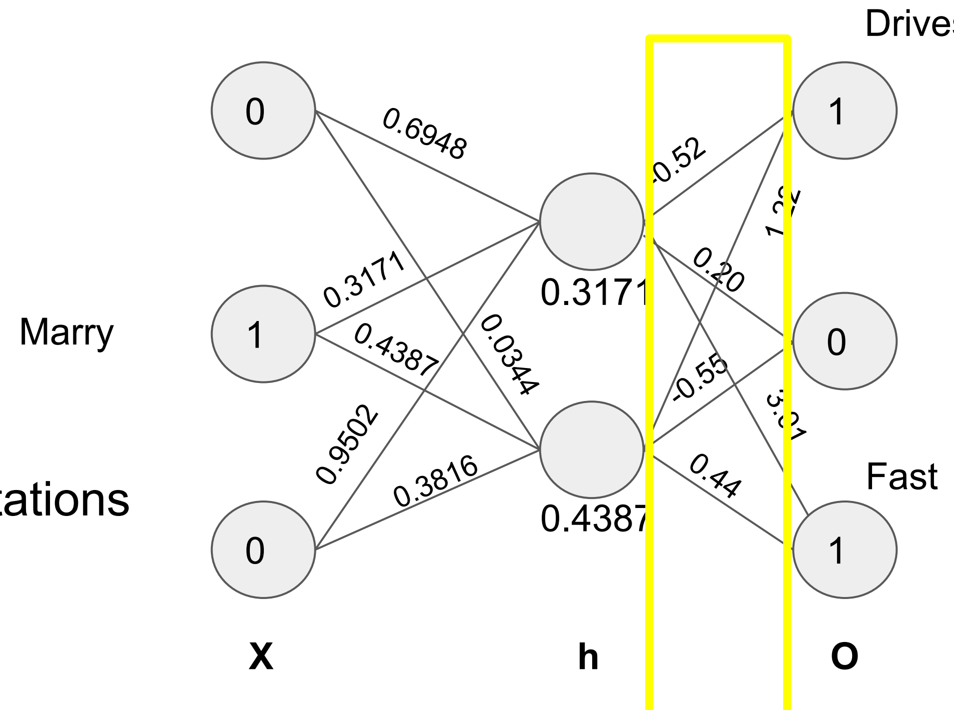
Wxh weights are the learned representations of the words in the vocabulary



Feed-forward neural network

Multilayer Perceptron / Skip-gram

Input size = 3
Output size = 3



Also a continuous representation of words (currently ignored)

Who weights are also learned representations of the words in the vocabulary

Feed-forward neural network

Multilayer Perceptron / <u>Skip-gram</u>

Input size = 3
Output size = 3

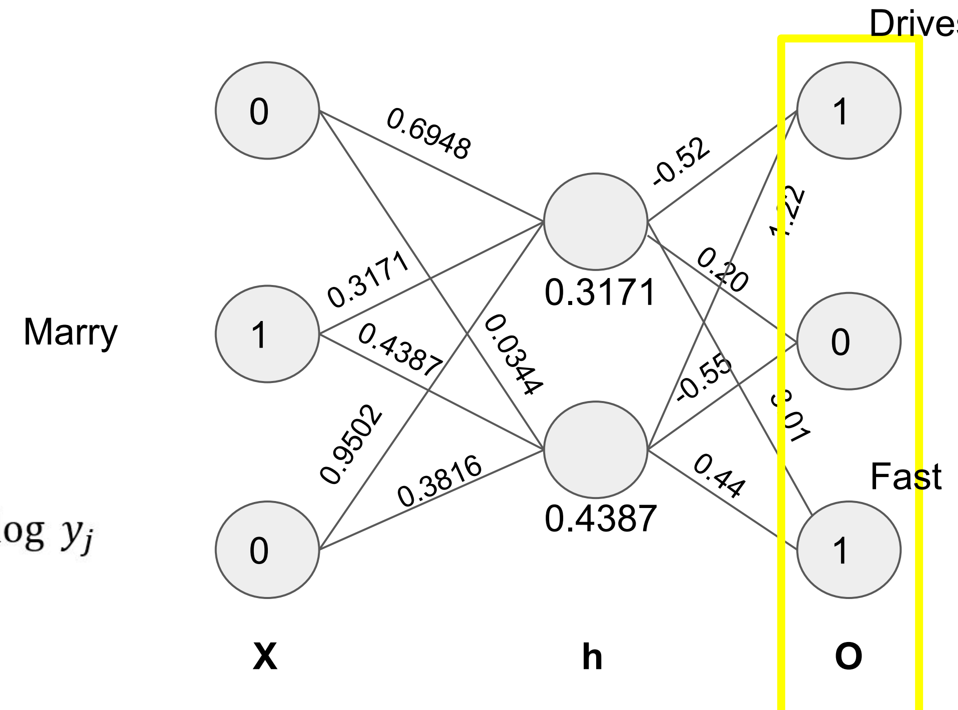
Activation: softmax

$$y_i = \frac{e^{z_i}}{\sum_{j \in Classes} e^{z_j}}$$

Loss: categorical cross-entropy

$$C = -\sum_{j \in Classes} t_j \log y_j$$

Alternative loss: binary cross-entropy for negative sampling variant

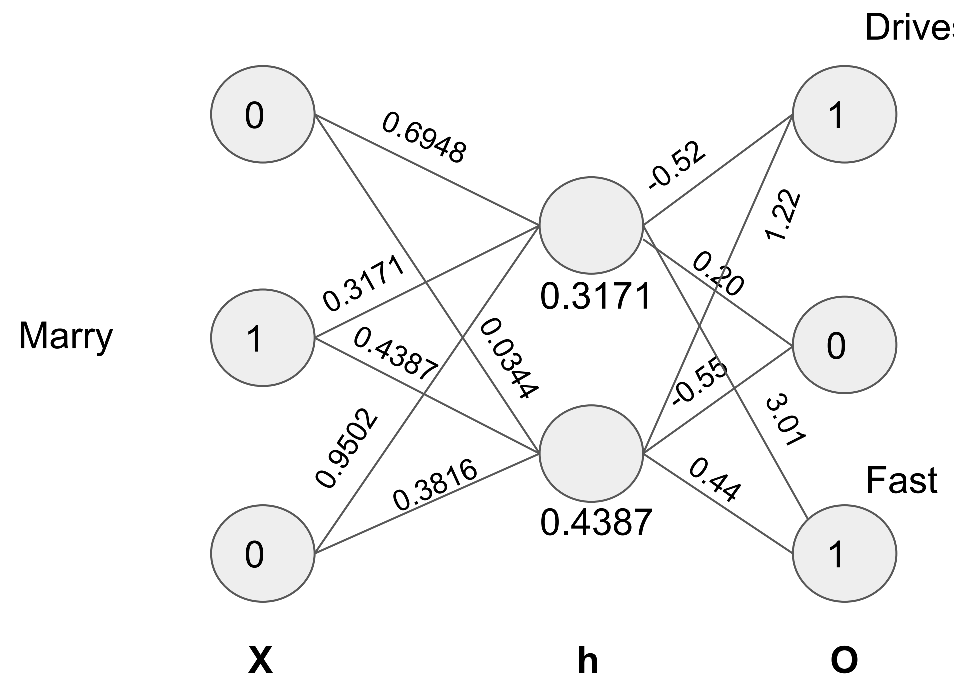


Feed-forward neural network

Multilayer Perceptron / Skip-gram

Input size = 3
Output size = 3

Skip-grams have a single word as input
context words as output



Feed-forward neural network

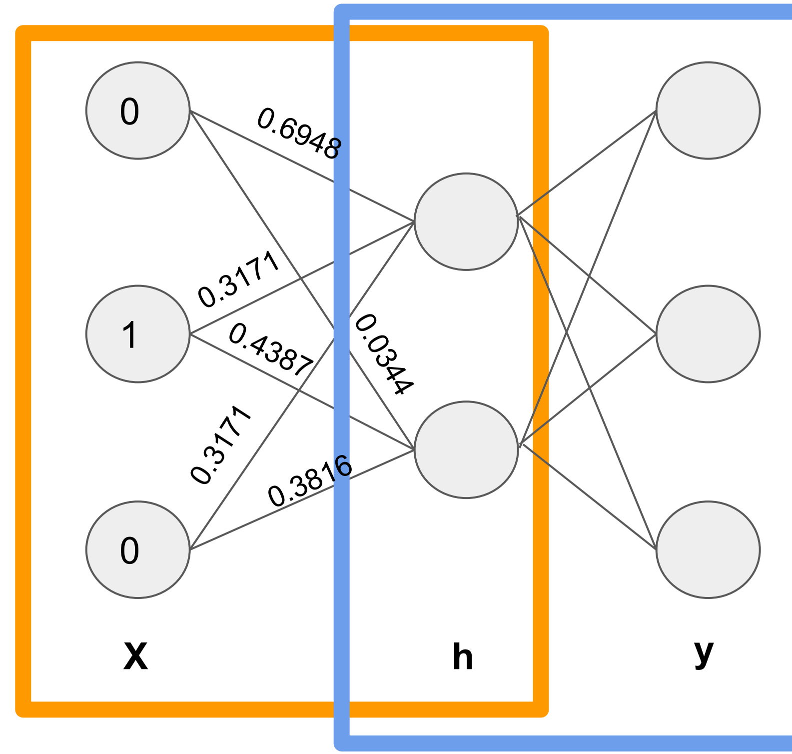
Multilayer Perceptron

Vocabulary size = 3
Nodes in hidden layer = 2

$h =$

$W_{hy} =$

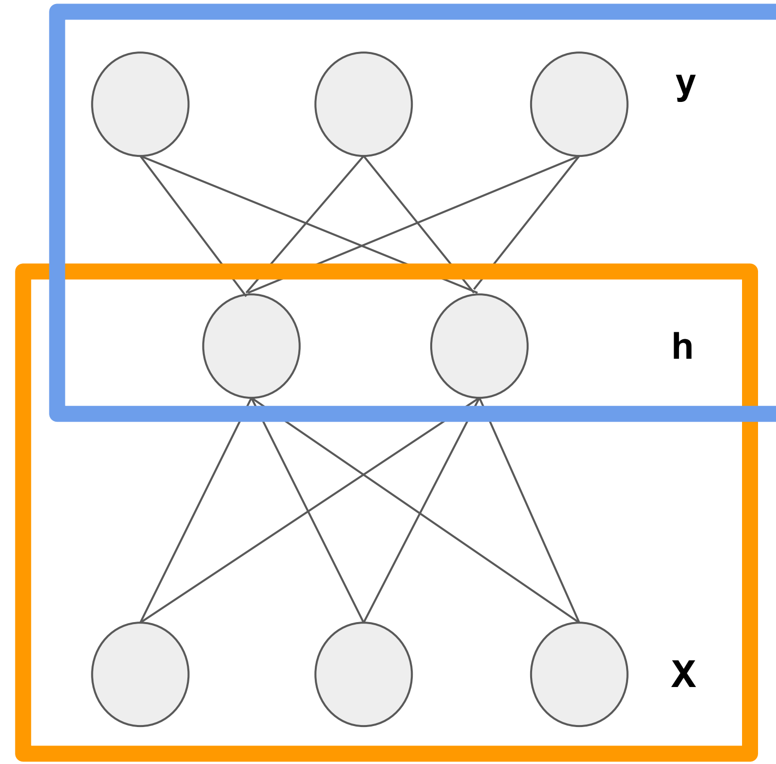
$y =$



Feed-forward neural network

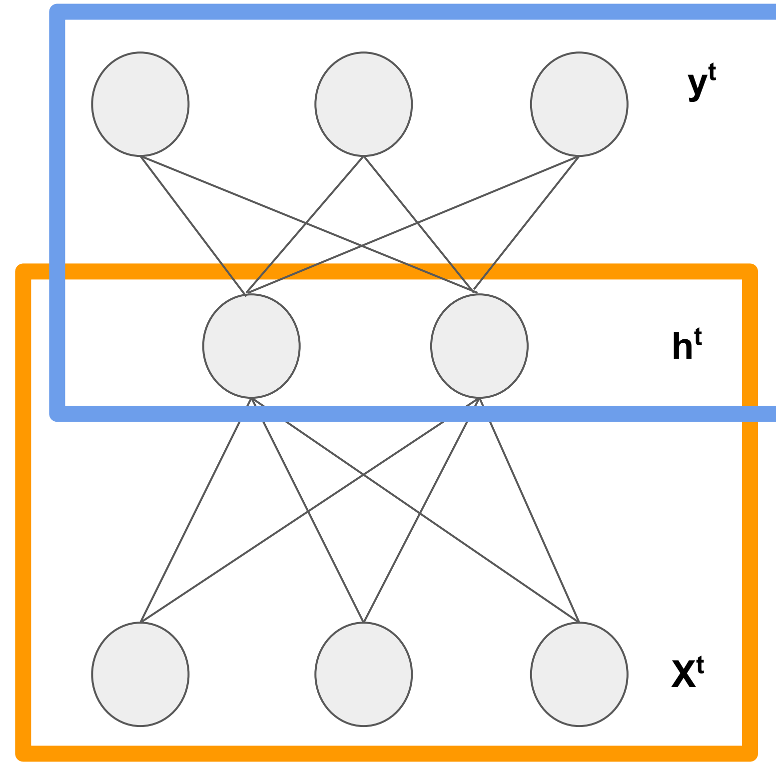
Multilayer Perceptron

Vocabulary size = 3
Nodes in hidden layer = 2



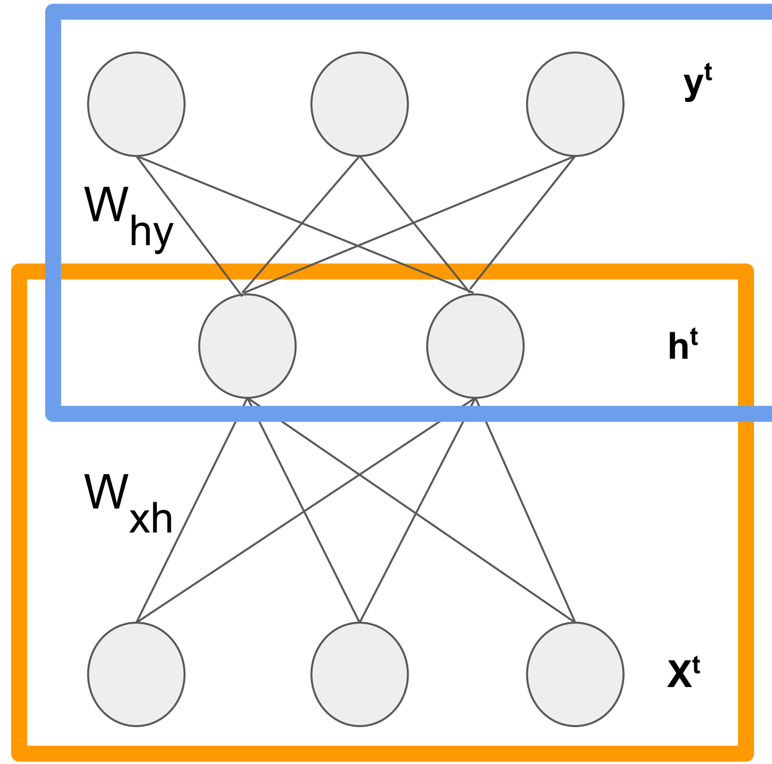
Time slice notation

Vocabulary size = 3
Nodes in hidden layer = 2



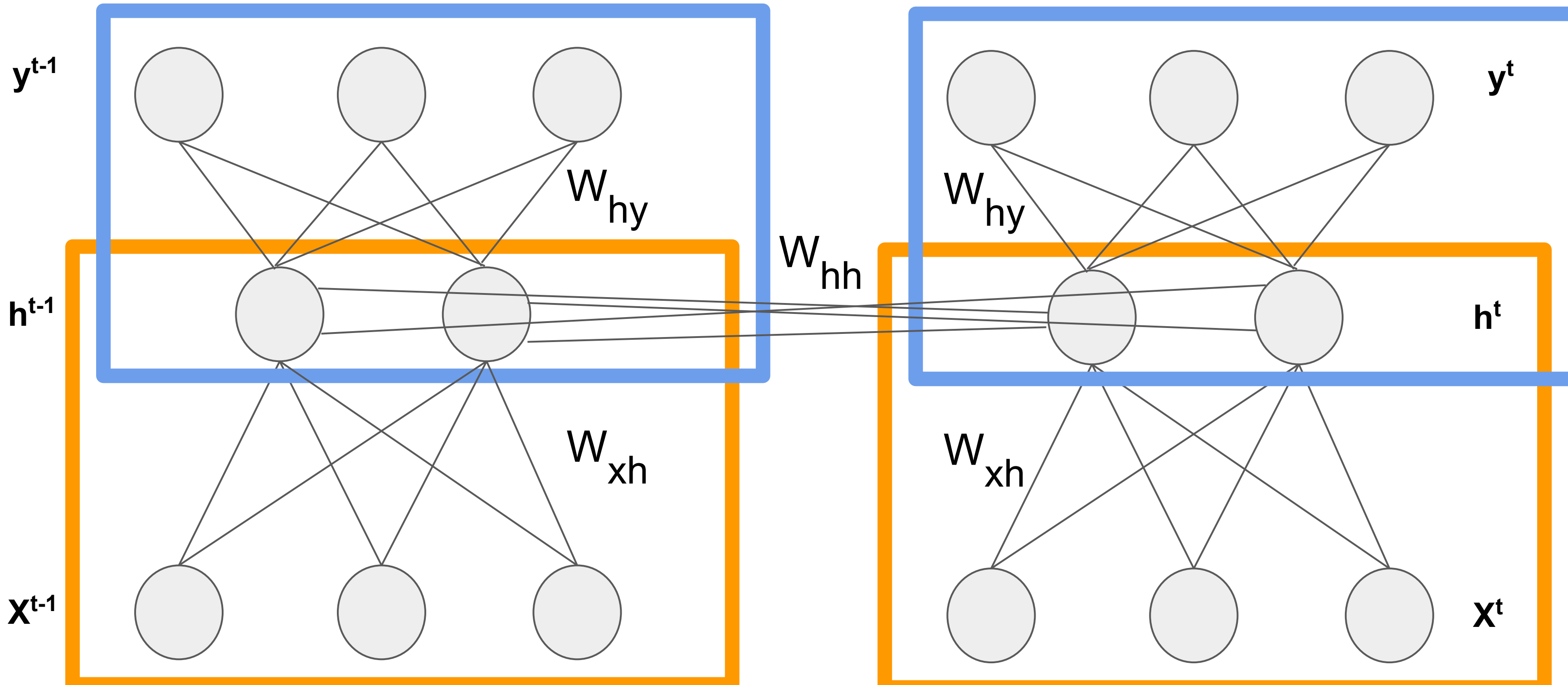
Input & Output weights

Vocabulary size = 3
Nodes in hidden layer = 2



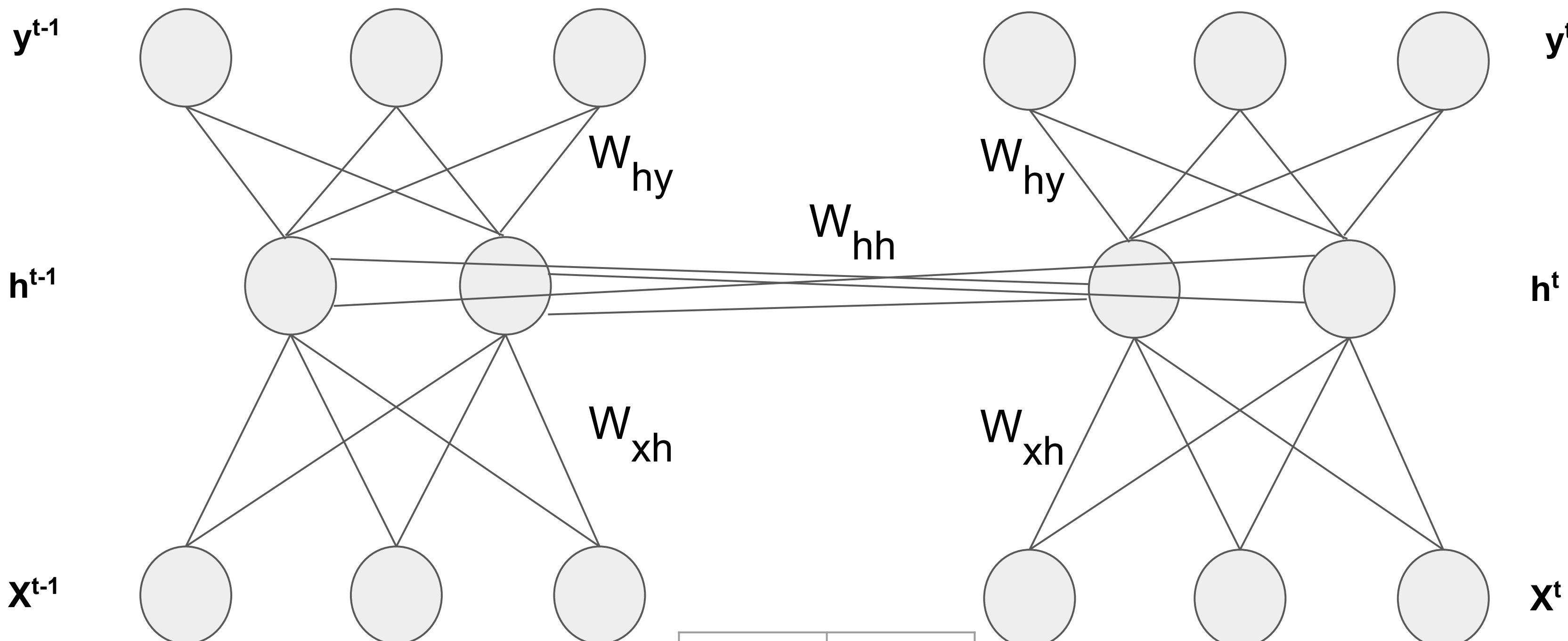
Recurrent Neural Network

Vocabulary size = 3
Nodes in hidden layer = 2



Recurrent Neural Network

Vocabulary size = 3
Nodes in hidden layer = 2

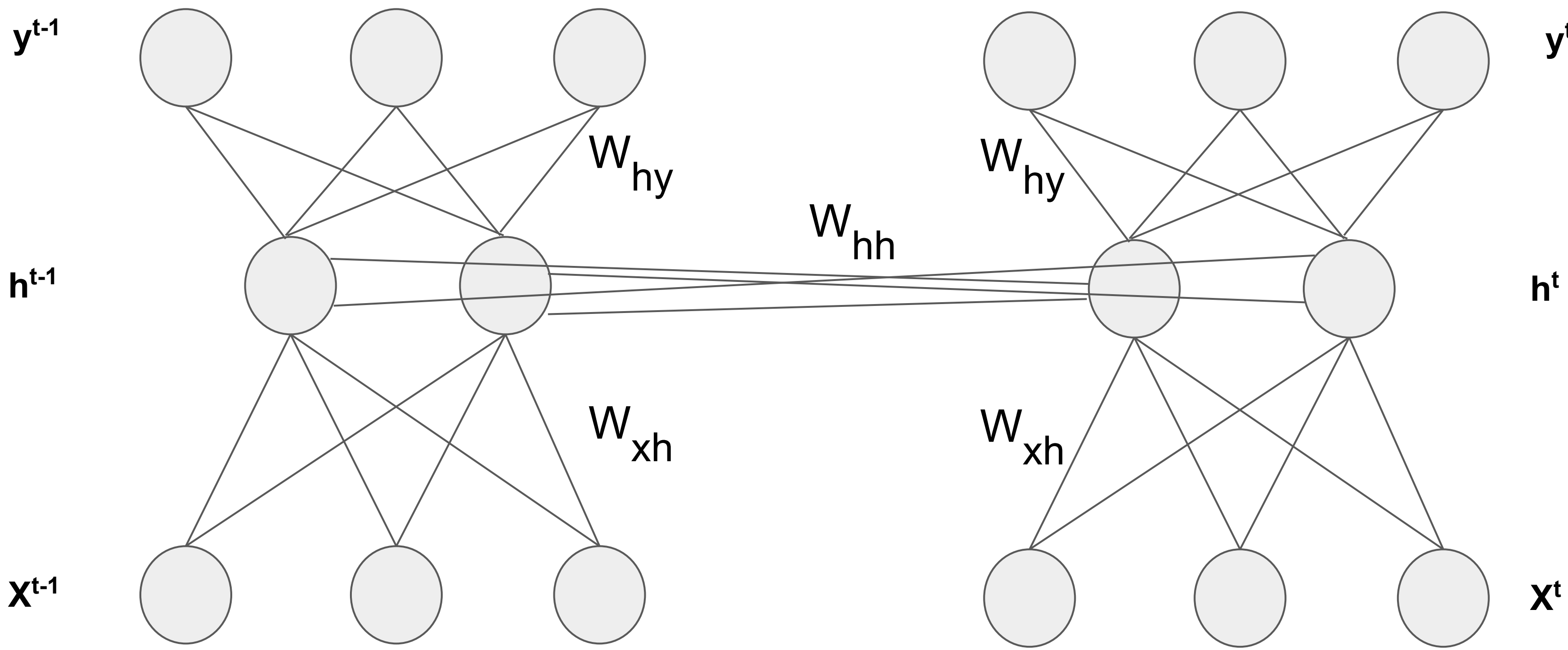


W_{hh} =

0.4522	0.9127
0.2232	0.9686

Recurrent Neural Network

Vocabulary size = 3
Nodes in hidden layer = 2



$$h^{(t)} = tanh(W_{hh}h^{(t-1)} + W_{xh}x^{(t)} + b_h)$$

Recurrent Neural Network

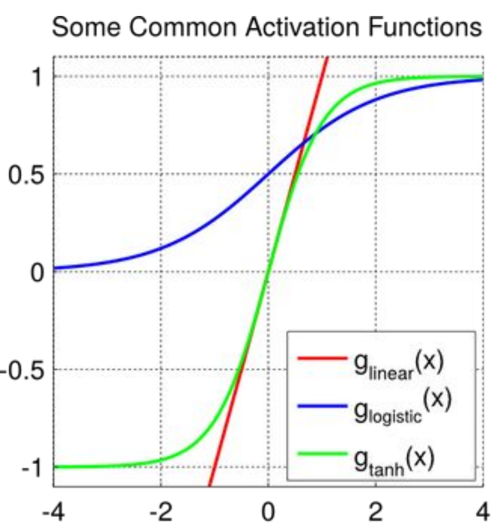
Same neural network principles apply

Loss

$$C = -\sum_{j \in Classes} t_j \log y_j$$

Cross-entropy loss

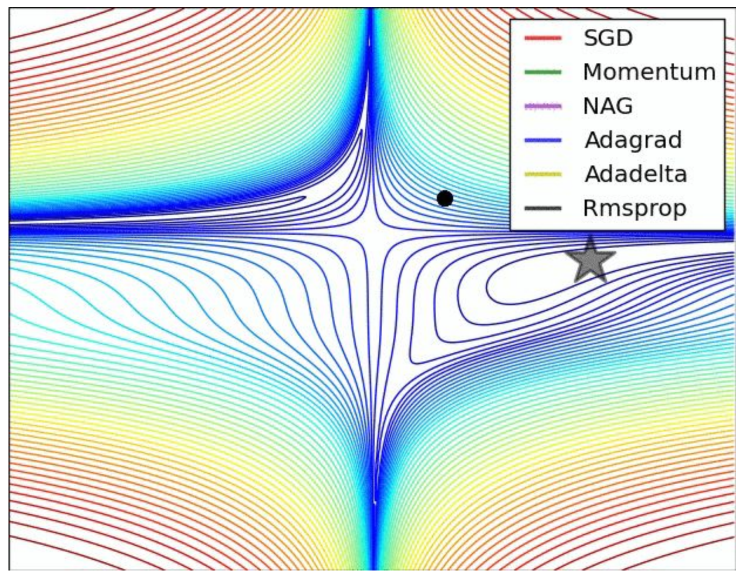
Activations



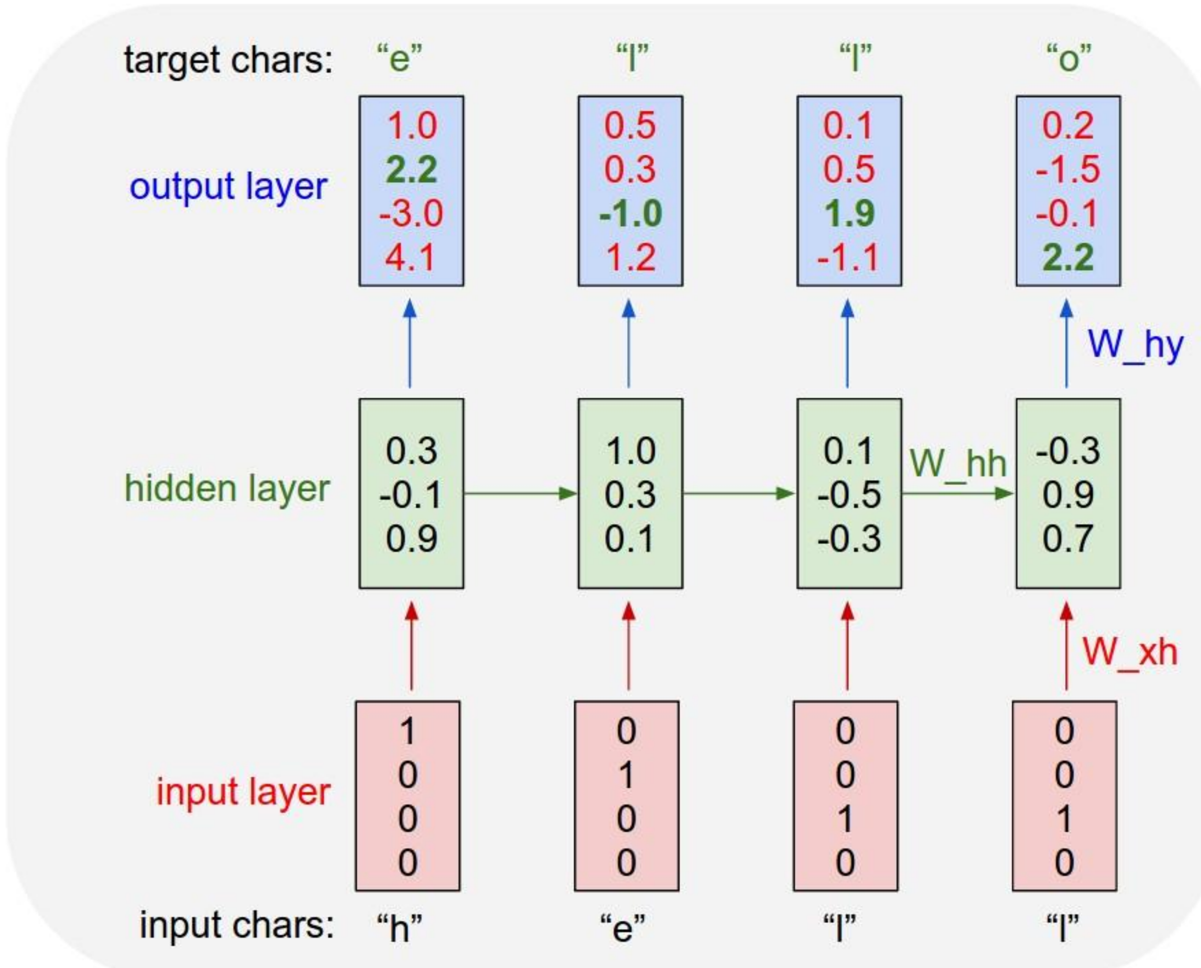
$$y_i = \frac{e^{z_i}}{\sum_{j \in Classes} e^{z_j}}$$

Softmax
Output Layer Activation
(for categorical outputs)

Backpropagation through time

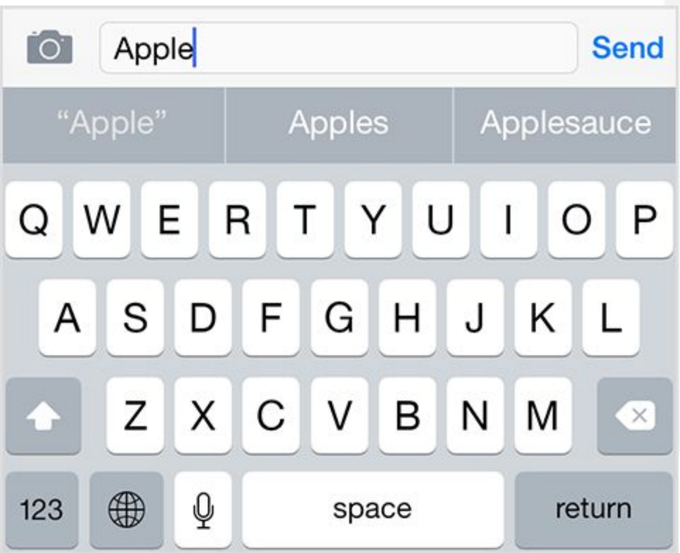


RNN Examples

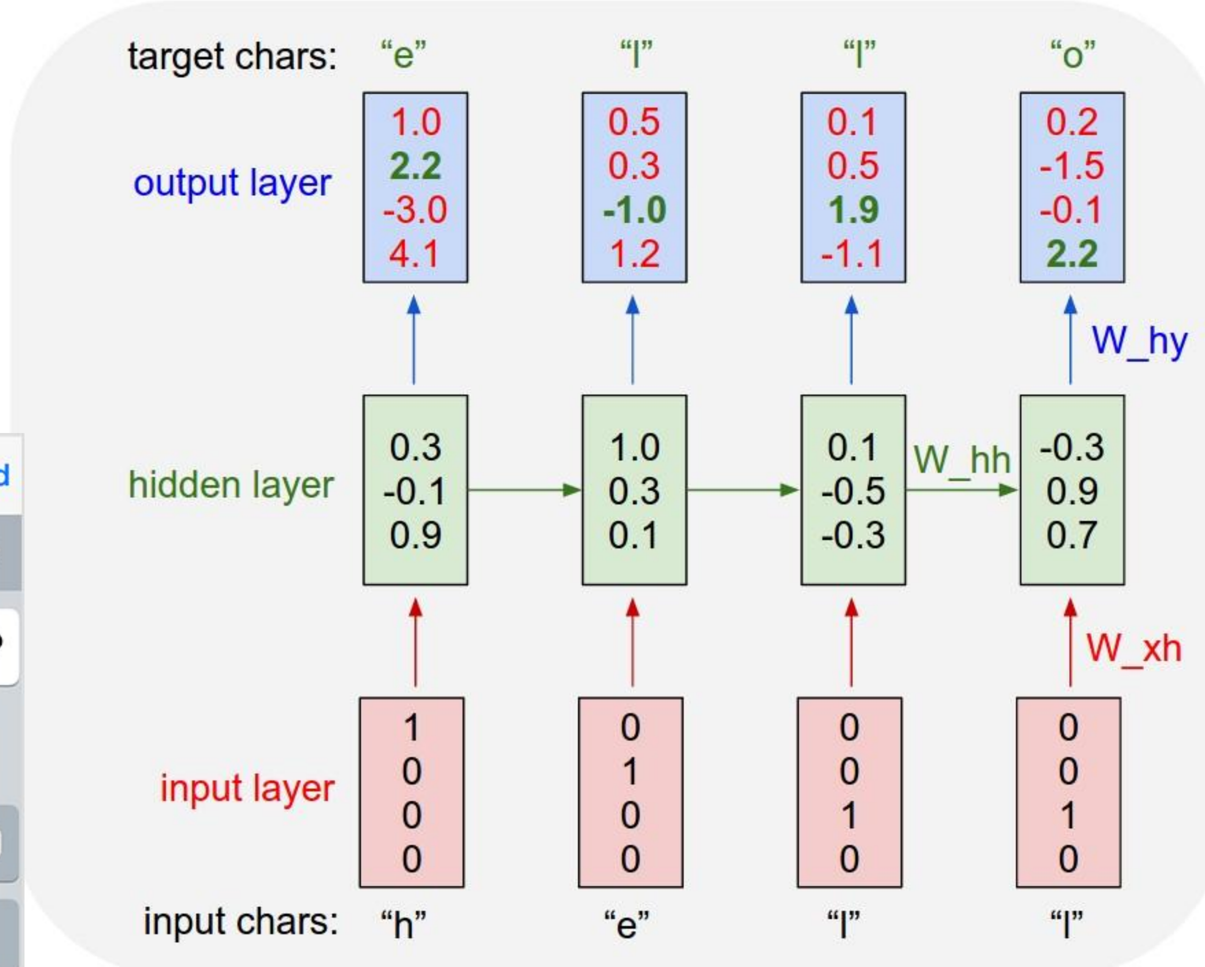


[paper]

RNN Examples



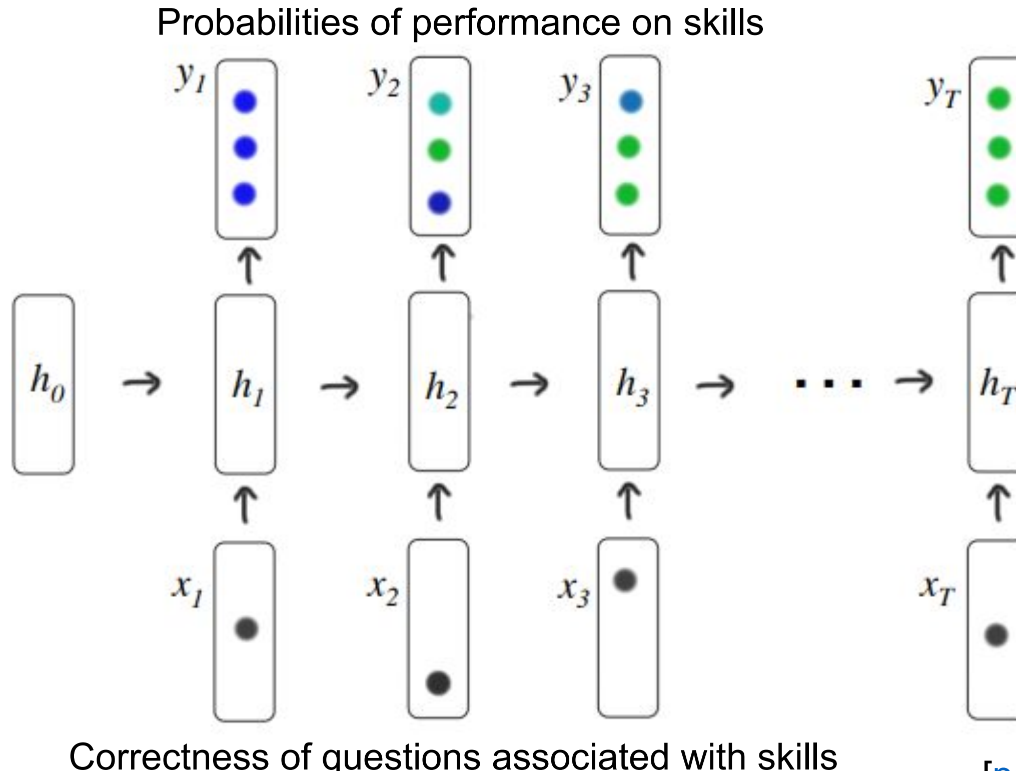
Used in predictive keyboards



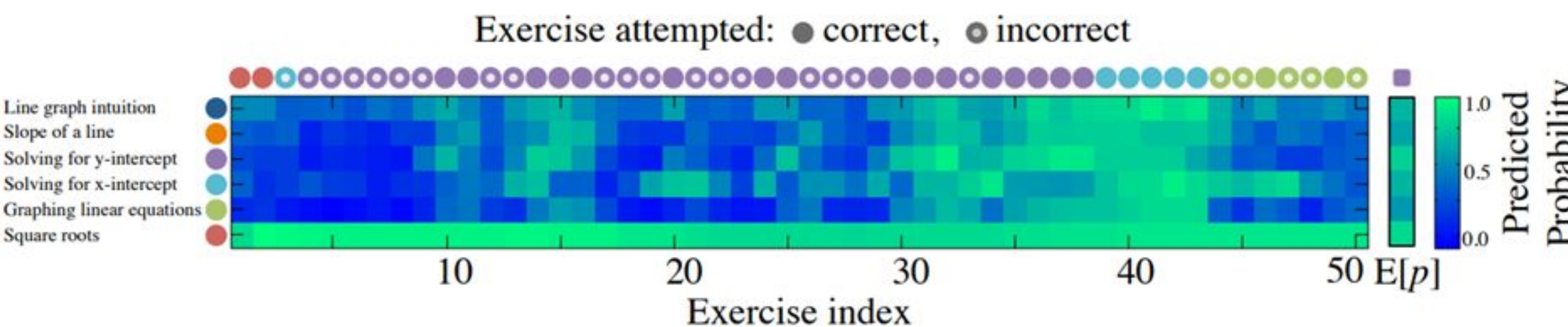
[paper]

RNN Examples

Problem solving in Khan Academy

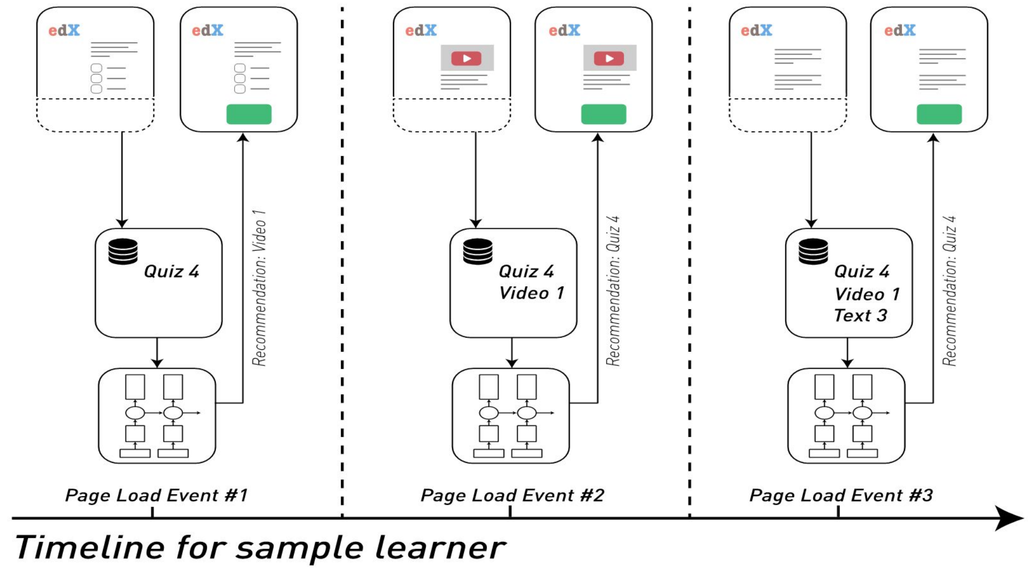


[paper]



RNN Examples

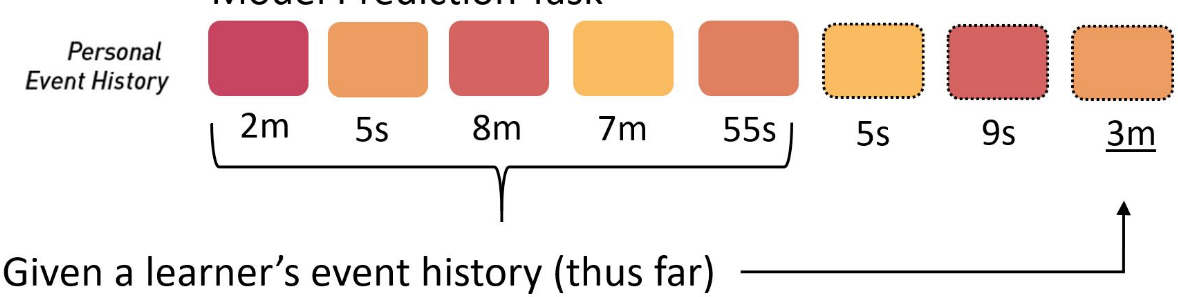
Personalized recommendation in an online course



[Demo video](#)

[[paper](#)]

Model Prediction Task

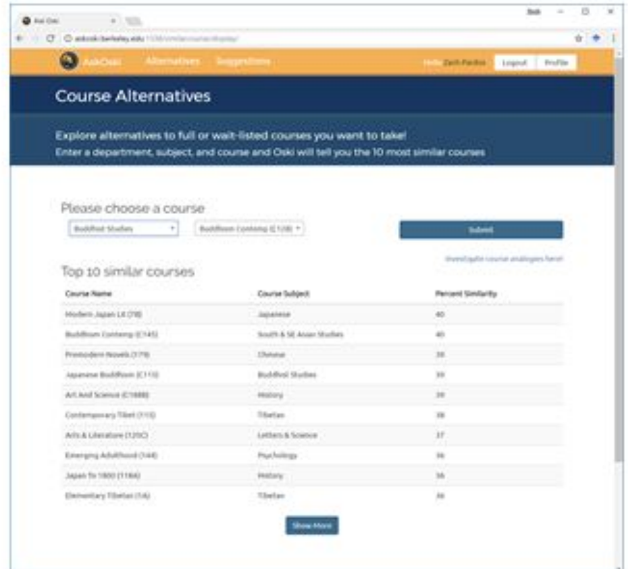


Keep predicting ahead until a resource is found where the learner is predicted to spend > 60s on

RNN Examples

Personalized course information at UCB

Surfacing information to students



(From enrollments)

- Course similarities
- Course relationships to other subjects
- Registrar's recommended list
- Degree requirements (under development)
- Positive student response
- Designing for the community college system to address issues in transfer student success

Acknowledgement: Andrew Eppig (OPA), Mark Chiang (EDW), Johanna Metzgar (OR), Jen Stringer (ETS), Aswan Mow (EDW), Daniel Grieb (EDW), and Max Michel (EDW), and Walter Wong (OR) for data assistance and documentation.

[talk]

Combines skip-grams and RNNs

https://askoski.berkeley.edu