Training & Explaining

Long Short Term Memory Models

Bill Gold @NYHACKR

Agenda

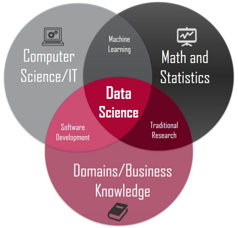
I. Why Explainability?

II. Training: LSTM an Overview

III. Explaining: Activiations and Predictions

about bill

+$500MM ROI



Hundreds of Models

+10 Platforms

Ataeva

HAYSTREAM

Columbia University in the City of New York

NYU

Georgia Tech

billcgold

linkedin.com/in / **billcgold**

github.com / **billcgold**

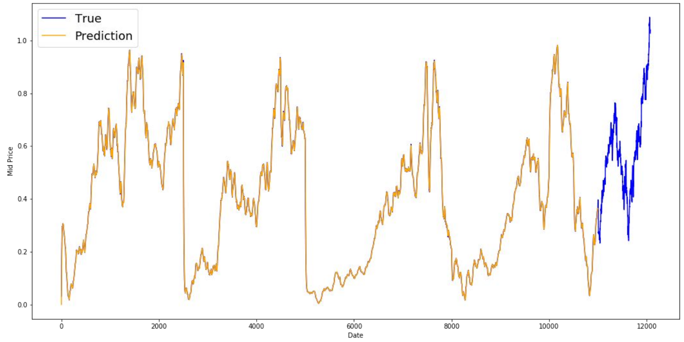
Look Here For Presentation & Code

billcgold @gmail.com

@**billcgold**

Why Explainability?

Sequences, a Broad View



- Numeric - Stock Prices
- Pixels - Images or Video
- Audio
- Text & Natural Language Processing (NLP)

https://www.datacamp.com/community/tutorials/lstm-python-stock-market

LSTM Models Are Used For ...

- Auto responses email, text messaging ...
- Document Creation, Summarization
- Customer Service Automation

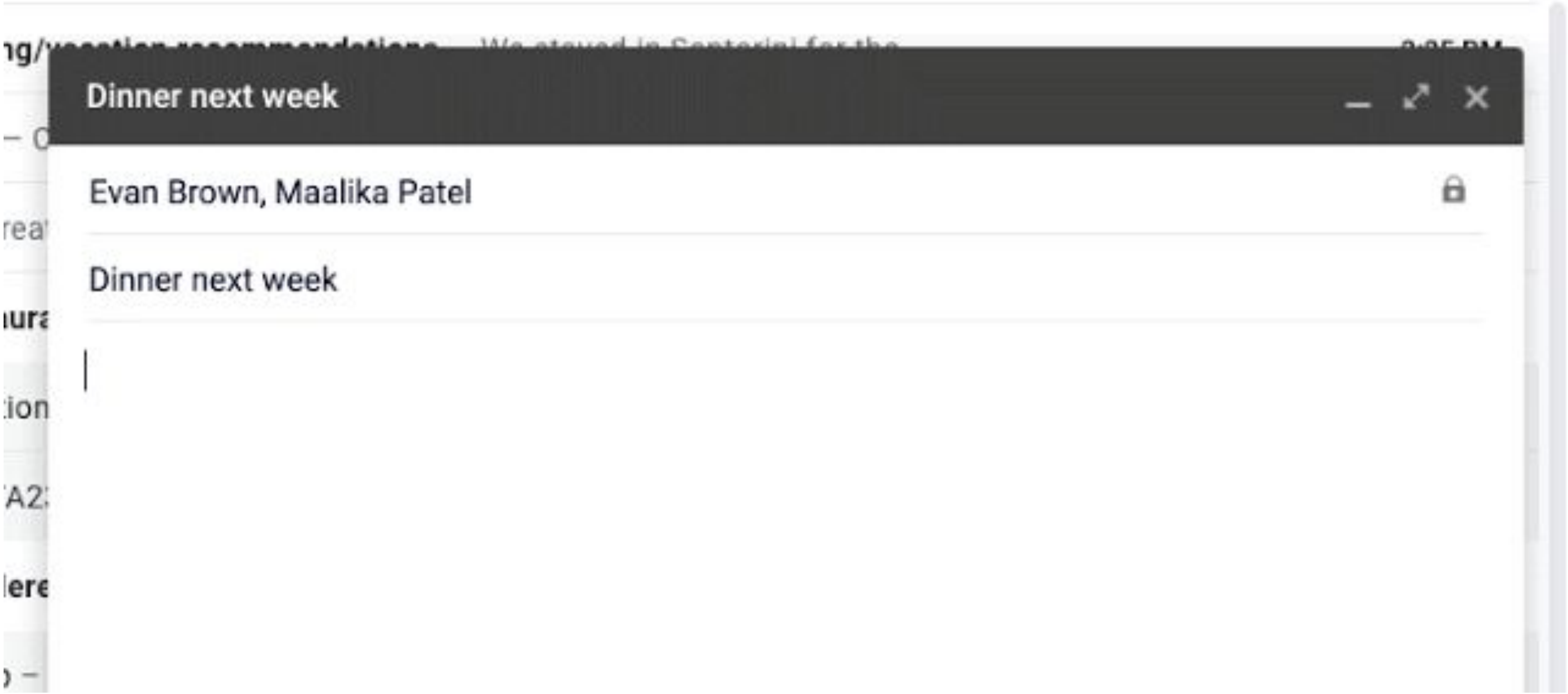
Example: Handwriting

Source:
Nikhil Buduma
LSTM RNN Demo

youtube.com/watch?v=mLxsbWAYIpw

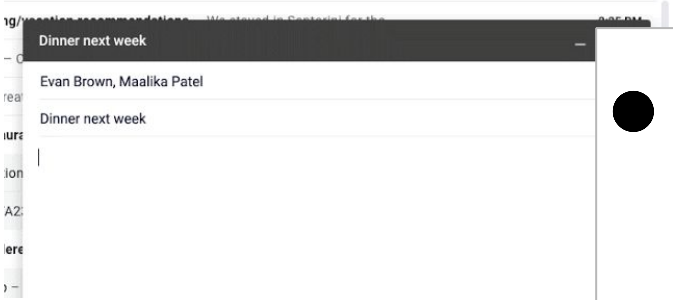
Smart Compose

Google AI



https://ai.googleblog.com/2018/05/smart-compose-using-neural-networks-to.html

Smart Compose



- Experimented with Many Methods
- seq2seq
- BoW & RNN-LM

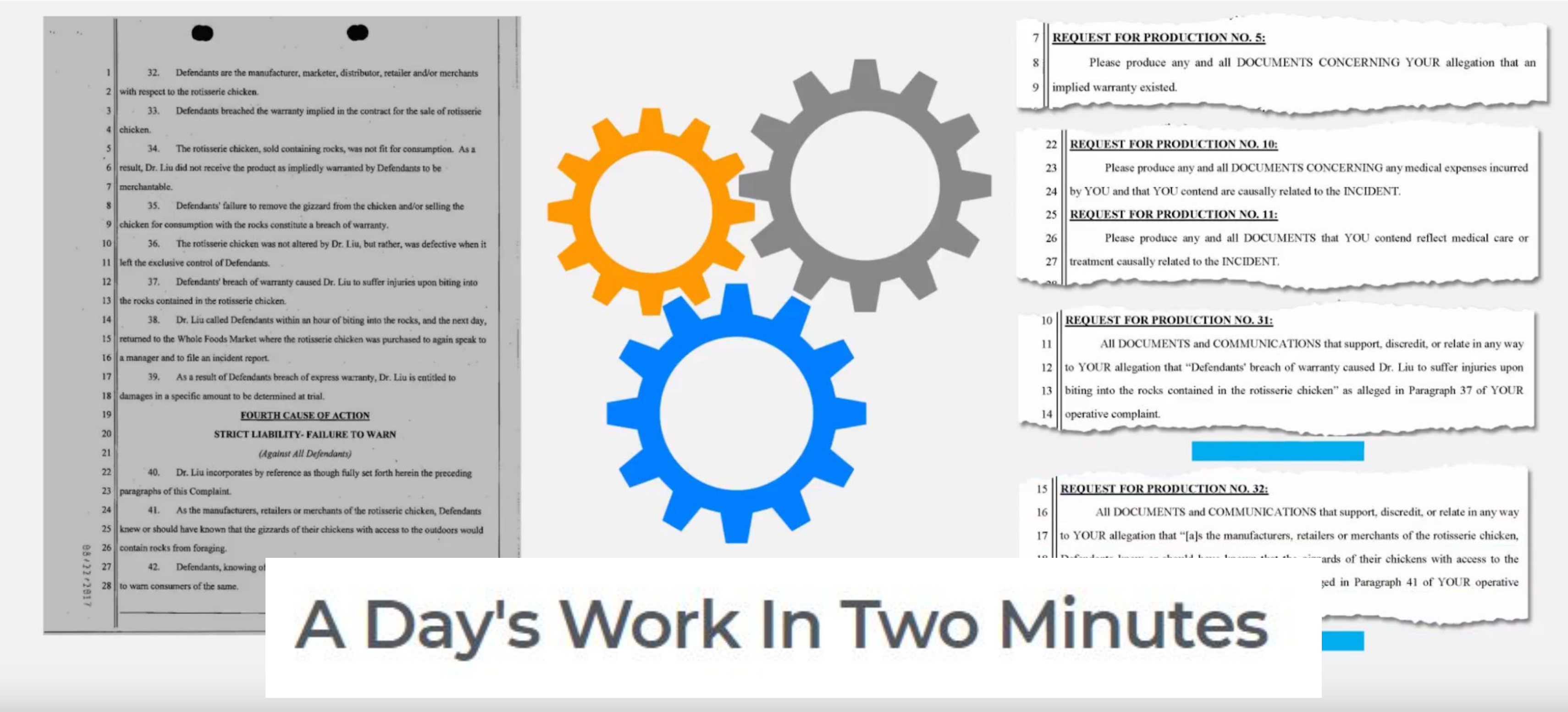
Google Blog
https://ai.googleblog.com/2018/05/smart-compose-using-neural-networks-to.html

Smart Compose

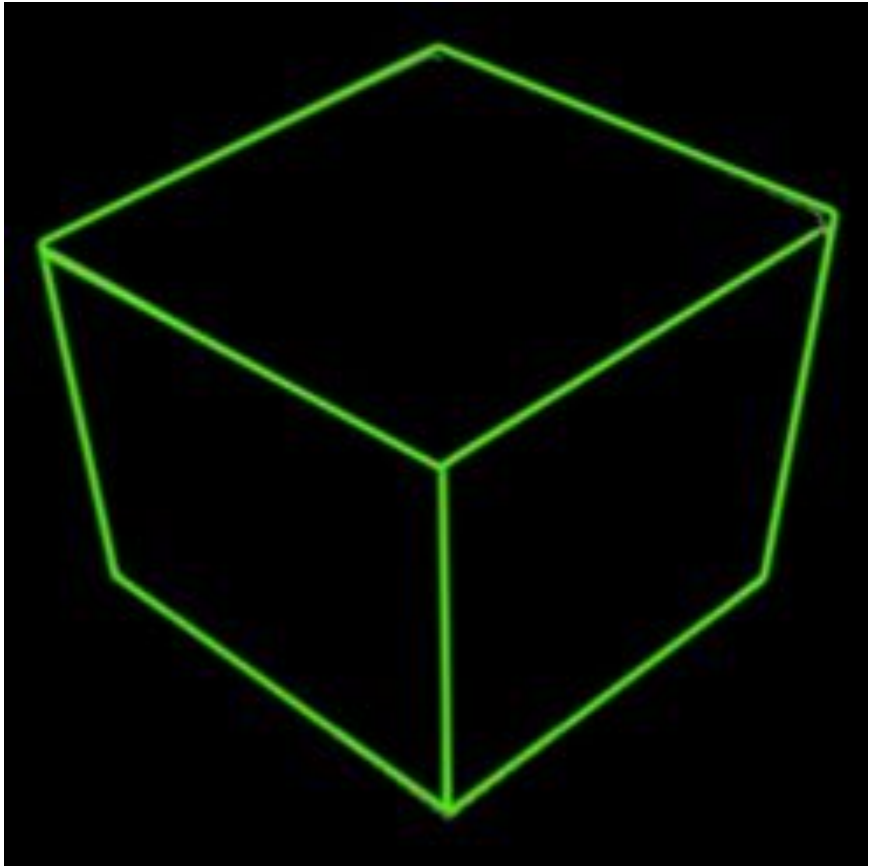
LEGALMATION®

Complaint

Requests for Production



Deep Learning has a Black Box Quality

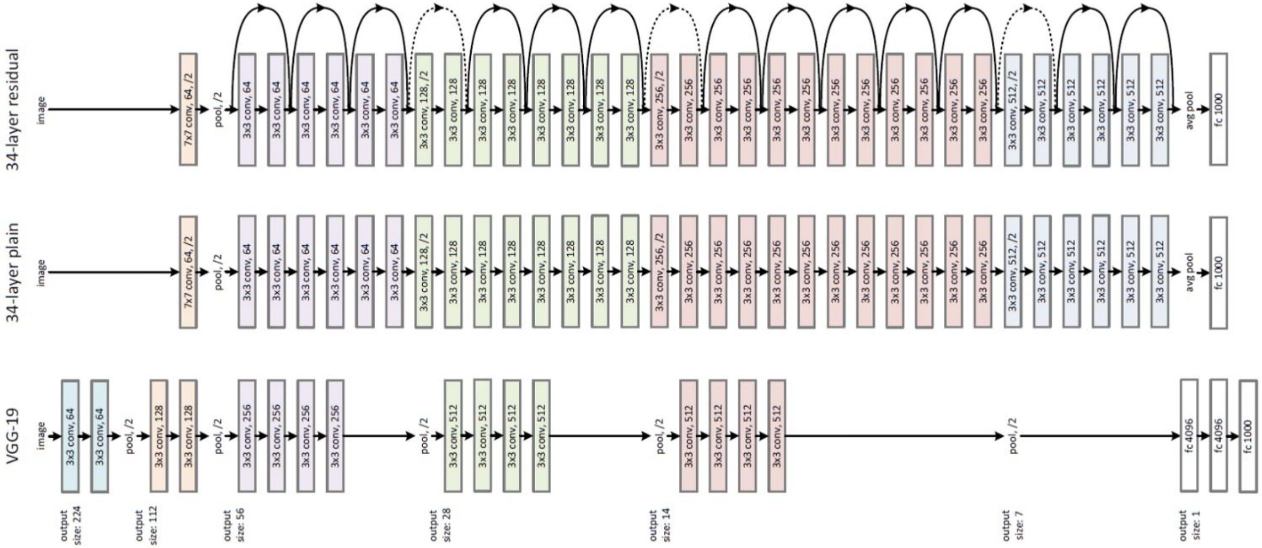


ResNet

Won a 2015 classification competition

Error rate of 3.57%.

Some ResNets' contain 152 layers



- Architecture
- Massive quantity of activations
- Unstructured data is a complicating factor

SIFI: 1, 5, 500, 25k

- 1 Model
- 5 Features
- 500 Pages of Documentation
- ~25k Team Hours (Guesstimate)
- x 15 Products

Who Values? Who is Compelled?

- Credit Risk - CCAR, Basel, CFPB
- Legal Community - Bias, Accuracy, Outliers, Intuitive
- Medical Diagnostics

Training LSTMs

Title Inspiration From …

CLYDEISMS


10 CLYDEISMS THIS SEASON (COURTESY CHI NWOGU AT BLOOMBERG SPORTS)
CLYDEISM
AVG USAGE PER GAME
DISHING AND SWISHING 1.20
SEE THE BALL, SEE YOUR MAN 0.85
MOVING AND GROOVING 0.78
POSTING AND TOASTING 0.76
WHEELING AND DEALING 0.76
SHAKING AND BAKING 0.72
SLICING AND DICING 0.63
HACKING AND WHACKING 0.57
STUMBLING AND BUMBLING 0.50
BOUNDING AND ASTOUNDING 0.41
WALT
FRAZIER

Training LTSMs

Goal of this Module is Intuition

- Architecture
- Mathematics
- Train & R Code

Recurrent Neural Networks (RNN)

- What is an RNN?
- “Unreasonable Effectiveness”
- Key Gaps
- Example

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

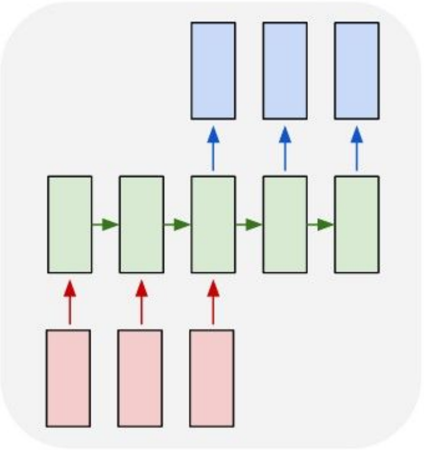
What is an RNN?

Output Layer

Hidden Layer

Input Layer

Sequence

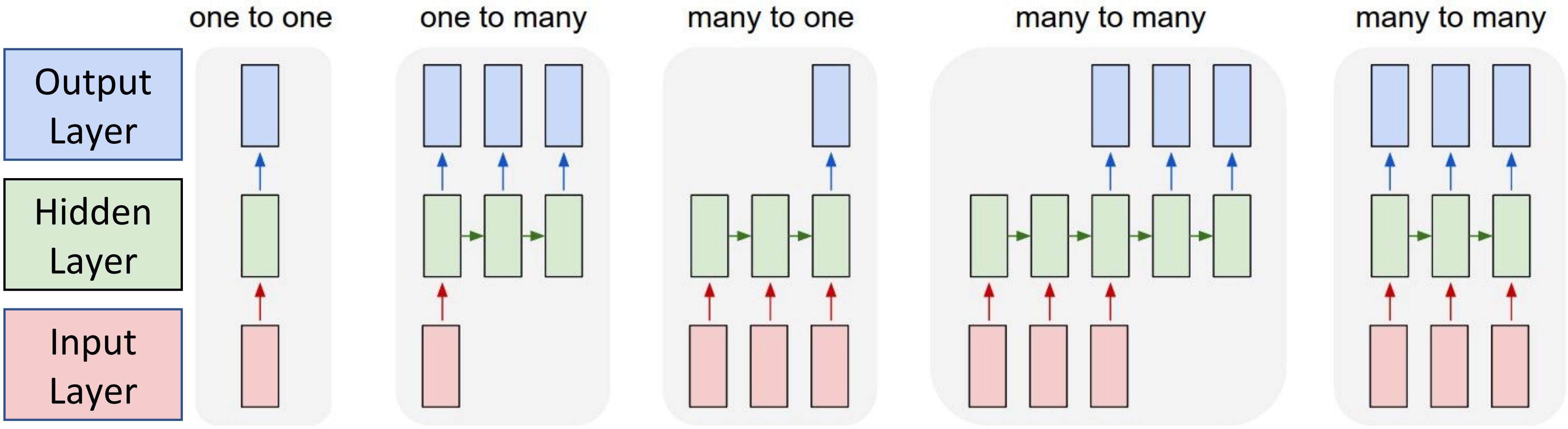


Sequence

Time Shift

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

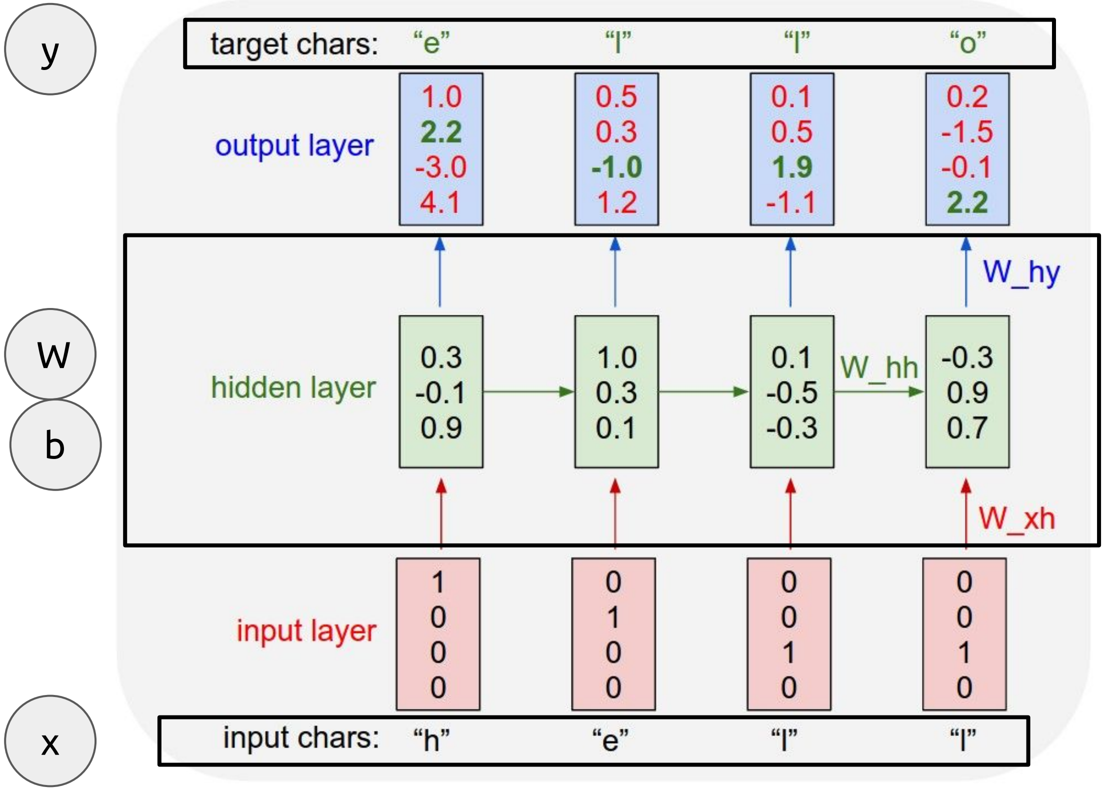
What is an RNN?



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

What is an RNN?

$$y = xW + b$$



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

“Unreasonable Effectiveness” of RNNs

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

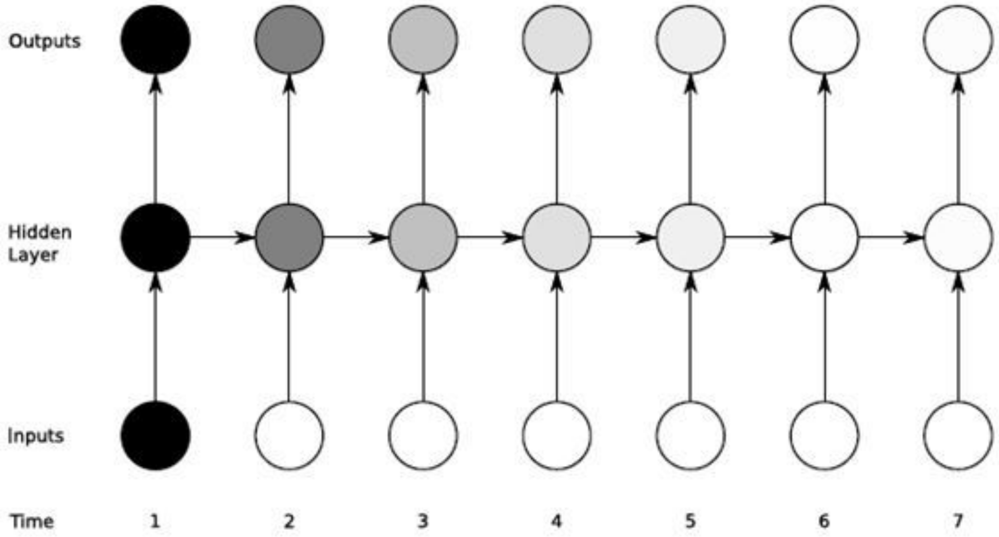
Key Gap - Exploding Gradient

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

Key Gap - Vanishing Gradient

I grew up near Queens. Great childhood memories are associate with sports, friends, family and my dog. My favorite baseball team is

Key Gap - Vanishing Gradient



$$a_h^t = \sum_{i=1}^{I} w_{ih} x_i^t + \sum_{h'=1}^{H} w_{h'h} b_{h'}^{t-1}$$

$$b_h^t = \theta_h(a_h^t)$$

Sensitivity decay exponentially over the time

Hyperparameters

I grew up near Queens. Great childhood memories are associate with sports, friends, family and my dog. My favorite baseball team is ____

Long Short Term Memory Hochreiter (et all)

LONG SHORT-TERM MEMORY

NEURAL COMPUTATION 9(8):1735–1780, 1997

Sepp Hochreiter
Fakultät für Informatik
Technische Universität München
80290 München, Germany
hochreit@informatik.tu-muenchen.de
http://www7.informatik.tu-muenchen.de/~hochreit

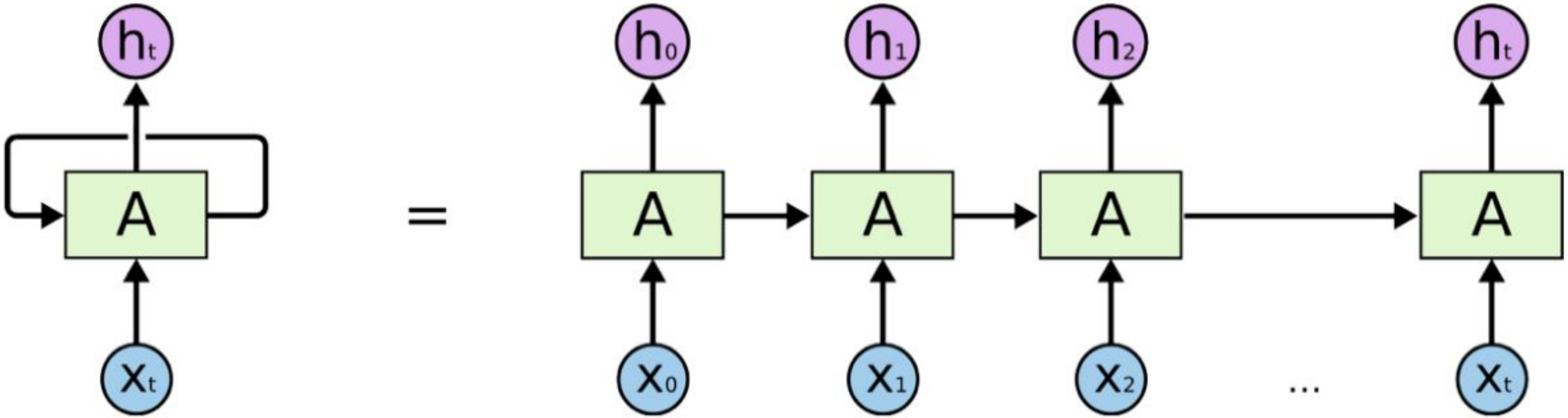
Jürgen Schmidhuber
IDSIA
Corso Elvezia 36
6900 Lugano, Switzerland
juergen@idsia.ch
http://www.idsia.ch/~juergen

Abstract

Learning to store information over extended time intervals via recurrent backpropagation takes a very long time, mostly due to insufficient, decaying error back flow. We briefly review Hochreiter's 1991 analysis of this problem, then address it by introducing a novel, efficient, gradient-based method called "Long Short-Term Memory" (LSTM). Truncating the gradient where this does not do harm, LSTM can learn to bridge minimal time lags in excess of 1000 discrete time steps by enforcing *constant* error flow through "constant error carrousels" within special units. Multiplicative gate units learn to open and close access to the constant error flow. LSTM is local in space and time; its computational complexity per time step and weight is $O(1)$. Our experiments with artificial data involve local, distributed, real-valued, and noisy pattern representations. In comparisons with RTRL, BPTT, Recurrent Cascade-Correlation, Elman nets, and Neural Sequence Chunking, LSTM leads to many more successful runs, and learns much faster. LSTM also solves complex, artificial long time lag tasks that have never been solved by previous recurrent network algorithms.

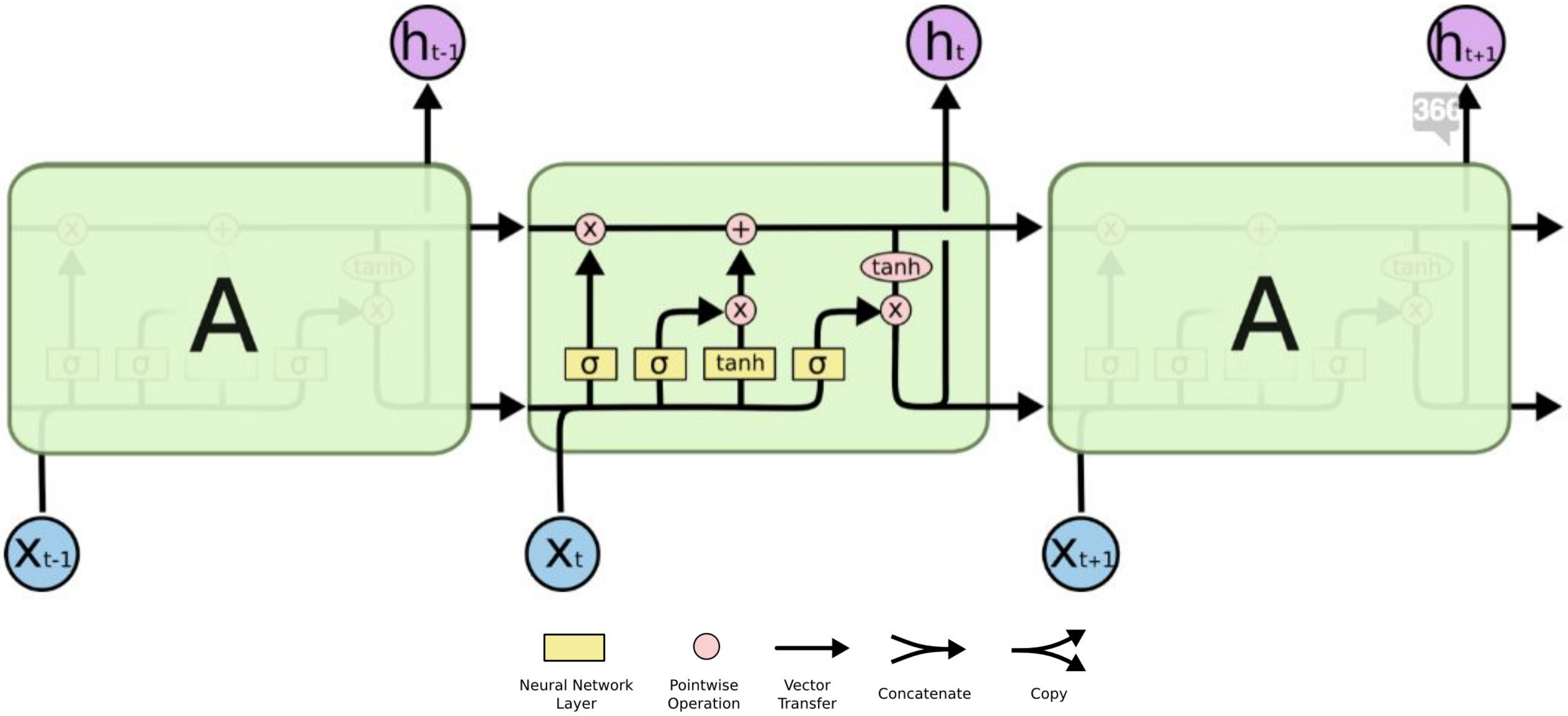
More effectively model long term interactions

What is an LSTM



"Understanding LSTM Networks" (2015)

What is an LSTM



"Understanding LSTM Networks" (2015)

Network within a Network

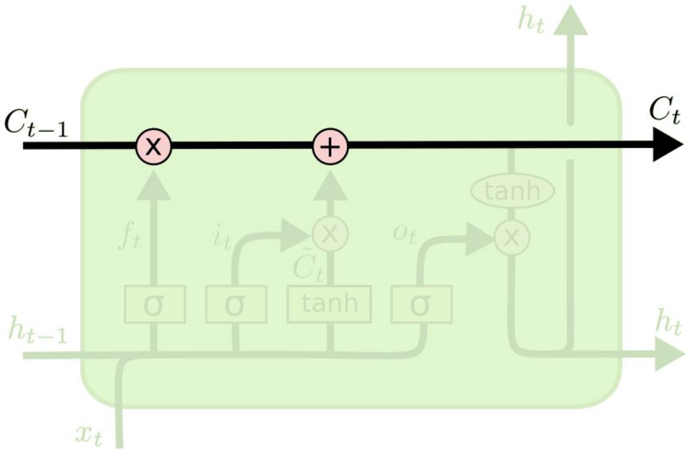
Unpack an LSTM

- Cell State (Remember)
- Forget
- Input
- Output

"Understanding LSTM Networks" (2015)

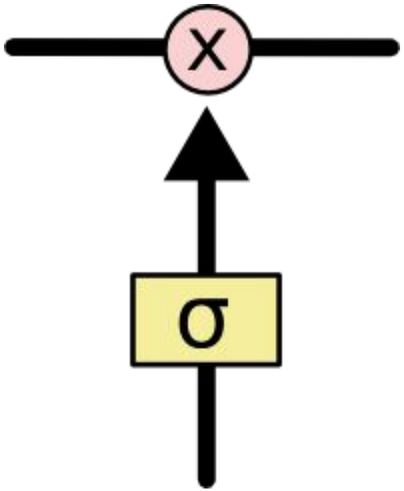
Cell State C_t

- Horizontal line running through the top
- Like a conveyor belt
- Some minor linear interactions.
- Easy for information to just flow unchanged



“Understanding LSTM Networks” (2015)

Gates



- Ability to remove or add information
- Regulated by structures called gates
- Sigmoid neural net layer
- A pointwise multiplication operation.

0 $\rightarrow$ Completely Throw Away
1 $\rightarrow$ Completely Retain

"Understanding LSTM Networks" (2015)

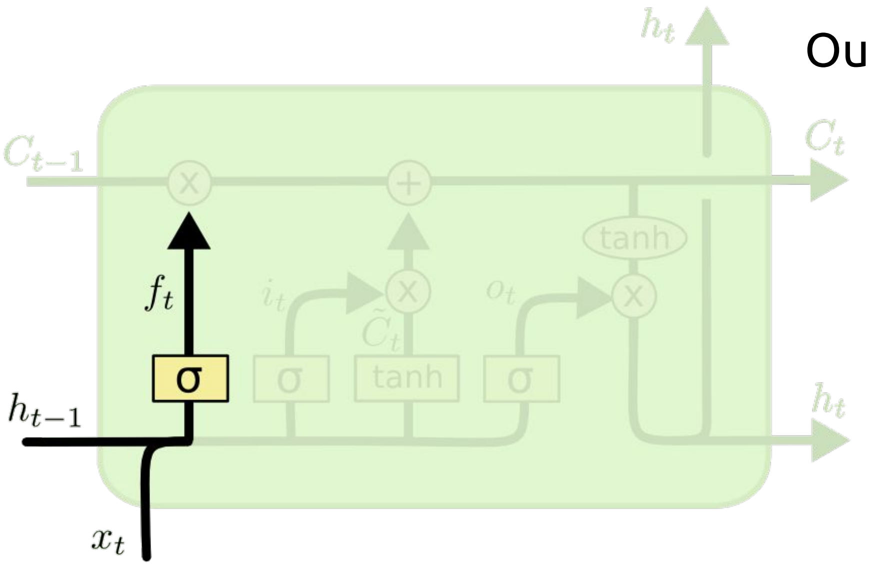
Gates


I went to the bank yesterday to talk about a loan
What do you need a loan for?
I don't. The bank does.

Forget f_t

Decide: What information we are going to throw away

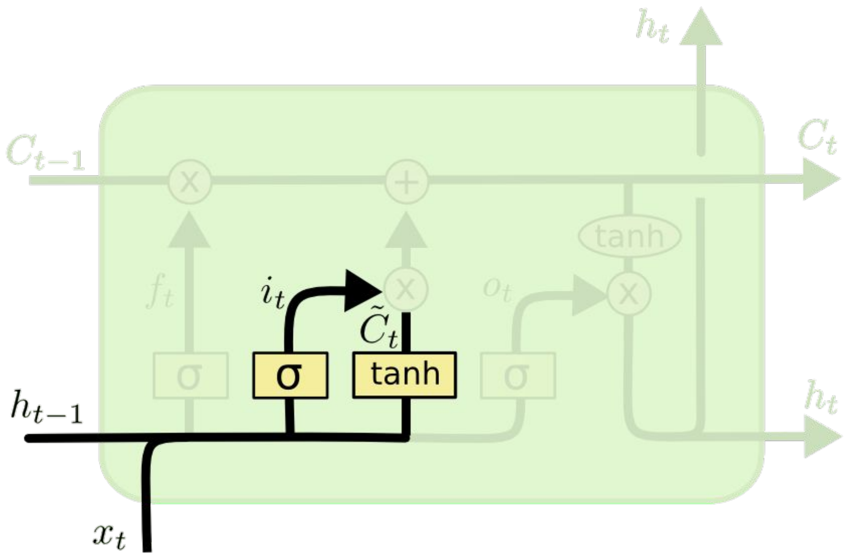
Outputs: 0 → Completely Throw Away
1 → Completely Retain



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

"Understanding LSTM Networks" (2015)

Input i_t and Candidate Cell State $\hat{C}_t$



Decide: Which values will be updated

Outputs: 0 → Completely Throw Away
1 → Completely Retain

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Decide: New Candidate Cell Values

Outputs: -1 → Completely Throw Away
1 → Completely Retain

"Understanding LSTM Networks" (2015)

Linguistic Intuition i_t and $\hat{C}_t$

Perhaps want to add gender of the new subject to the cell state, to replace the old one we're forgetting.

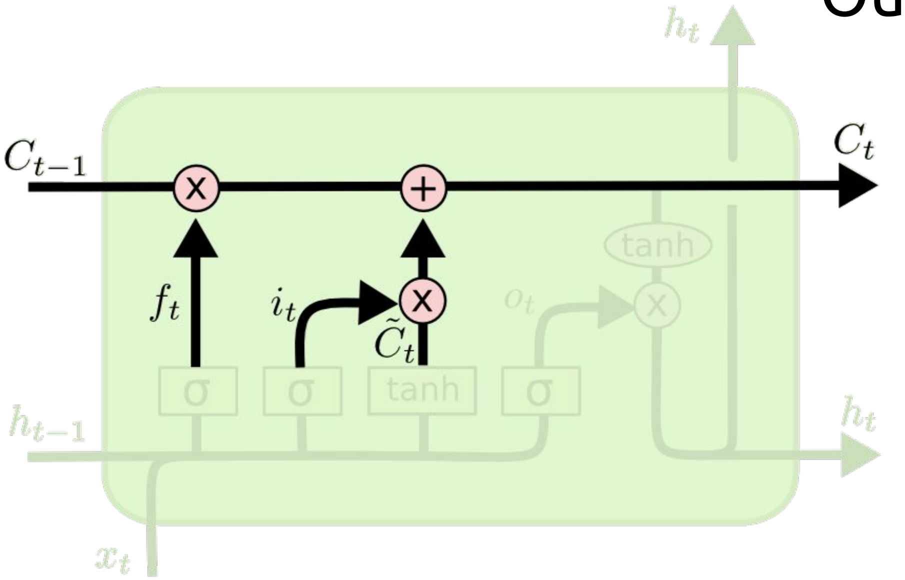
"Understanding LSTM Networks" (2015)

Cell State C_t

Cell State C_t

Action: Pointwise multiplication & addition operations

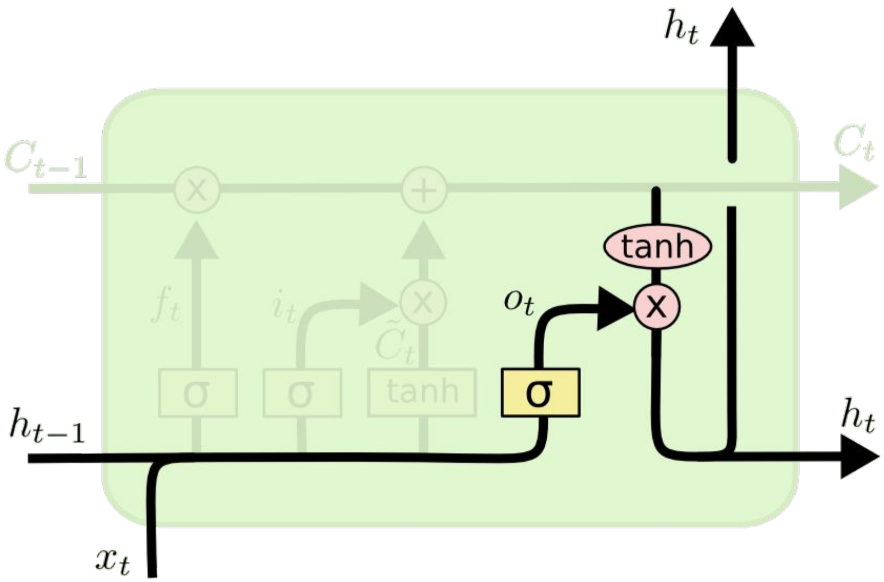
Outputs: C_t



$$C_t = \underbrace{\boxed{f_t * C_{t-1}}}_{\text{Forget}} + \underbrace{\boxed{i_t * \tilde{C}_t}}_{\text{Input \& Candidate Cell State}}$$

"Understanding LSTM Networks" (2015)

Output O_t



Decide: Which portions of cell we will output

Outputs: 0 → Completely Throw Away
1 → Completely Retain

$$o_t = \sigma\left(W_o \left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

"Understanding LSTM Networks" (2015)

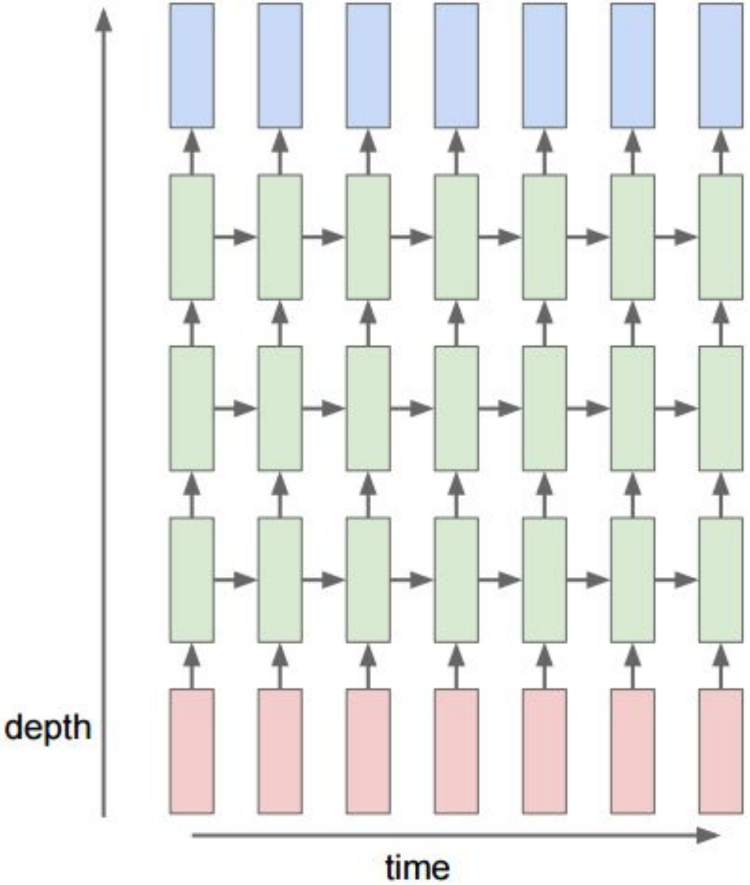
Linguistic Intuition Output O_t

- Just saw a subject,
- Perhaps connect a relevant verb
- Perhaps, it might output whether the subject is singular or plural for conjugation

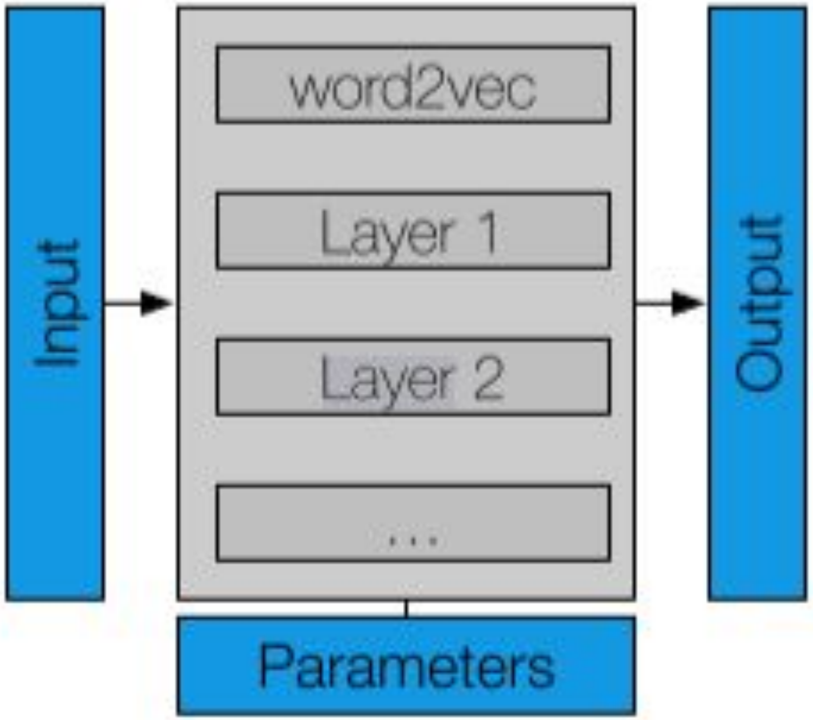
"Understanding LSTM Networks" (2015)

Layers (Deep)

Layers



Layers



Layers

hyper-parameters

- size
- number of layers
- model type
- batch size
- sequence length
- number of epochs
- clip gradients value
- learning rate
- decay rate
- gpu memory

Train an LSTM

Classify Questions - Sincere (0) or Not (1)

question_text	target
Do you have an adopted dog, how would you encourage people to adopt and not shop?	0
Is it crazy if I wash or wipe my groceries off? Germs are everywhere.	0
Can we use our external hard disk as a OS as well as for data storage.will the data be affected?	0
Why are Americans, British, Canadians, Australians and New Zealanders considered to be separate nations even when they all speak the same language?	1
If both Honey Singh and Justin Bieber fall from the 5th floor, who will survive?	1
Why don't poor countries print more money to use for paying for education, etc.?	1

Tom Aindow https://www.kaggle.com/taindow

Libraries & Parameters

```r
library(keras) # deep learning with keras

library(tidyverse) # importing, cleaning, visualising
library(tidytext) # working with text
library(data.table) # fast csv reading

options(scipen=999) # turn off scientific display


# Setup some parameters

max_words <- 15000 # Maximum number of words to consider as features
maxlen <- 64 # Text cutoff after n words
```

```
── Attaching packages ──────────────────────────────────────── tidyverse 1.2.1 ──
✔ ggplot2 2.2.1.9000     ✔ purrr   0.2.4
✔ tibble  1.4.2          ✔ dplyr   0.7.4
✔ tidyr   0.8.0          ✔ stringr 1.3.0
✔ readr   1.2.0          ✔ forcats 0.3.0
── Conflicts ─────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
✖ dplyr::vars()   masks ggplot2::vars()
```

https://www.kaggle.com/taindow

Create Tokens

```
train <- fread('../input/train.csv', data.table = FALSE)
test <- fread('../input/test.csv', data.table = FALSE)

test %>% head()
```

```
# Prepare to tokenize the text

full <- rbind(train %>% select(question_text), test %>% select(question_text))
texts <- full$question_text

# Tokenize - i.e. convert text into a sequence of integers

tokenizer <- text_tokenizer(num_words = max_words) %>%
  fit_text_tokenizer(texts)


sequences <- texts_to_sequences(tokenizer, texts)
word_index <- tokenizer$word_index

# Pad out texts so everything is the same length

data = pad_sequences(sequences, maxlen = maxlen)
```


Tom Aindow https://www.kaggle.com/taindow

Split Data

```
# Split back into train and test

train_matrix = data[1:nrow(train),]
test_matrix = data[(nrow(train)+1):nrow(data),]


# Prepare training labels

labels = train$target


# Prepare a validation set

set.seed(1337)

training_samples = nrow(train_matrix)*0.90
validation_samples = nrow(train_matrix)*0.10

indices = sample(1:nrow(train_matrix))
training_indices = indices[1:training_samples]
validation_indices = indices[(training_samples + 1): (training_samples + validation_samples)]

x_train = train_matrix[training_indices,]
y_train = labels[training_indices]

x_val = train_matrix[validation_indices,]
y_val = labels[validation_indices]

# Training dimensions

dim(x_train)
table(y_train)
```

```
y_train
      0        1
1102852    72657
```

Embeddings

```
lines <- readLines('../input/embeddings/wiki-news-300d-1M/wiki-news-300d-1M.vec')

fastwiki_embeddings_index = new.env(hash = TRUE, parent = emptyenv())

lines <- lines[2:length(lines)]

pb <- txtProgressBar(min = 0, max = length(lines), style = 3)
for (i in 1:length(lines)){
  line <- lines[[i]]
  values <- strsplit(line, " ")[[1]]
  word<- values[[1]]
  fastwiki_embeddings_index[[word]] = as.double(values[-1])
  setTxtProgressBar(pb, i)
}

fastwiki_embedding_dim = 300                                                # Create our embedding matrix
fastwiki_embedding_matrix = array(0, c(max_words, fastwiki_embedding_dim))

for (word in names(word_index)){
  index <- word_index[[word]]
  if (index < max_words){
    fastwiki_embedding_vector = fastwiki_embeddings_index[[word]]
    if (!is.null(fastwiki_embedding_vector))
      fastwiki_embedding_matrix[index+1,] <- fastwiki_embedding_vector # Words without an embedding are all zeros
  }
}

gc()
```


Tom Aindow https://www.kaggle.com/taindow

Embeddings

used	(Mb)	gc trigger	(Mb)	max used	(Mb)	
Ncells	11912023	636.2	17371378	927.8	17371378	927.8
Vcells	767284417	5854.0	1311293864	10004.4	807671889	6162.1

Tom Aindow https://www.kaggle.com/taindow

Model Architecture

```
input <- layer_input(                                      # Setup input
  shape = list(NULL),
  dtype = "int32",
  name = "input"
)

# Model layers

embedding <- input %>%
    layer_embedding(input_dim = max_words, output_dim = fastwiki_embedding_dim, name = "embedding")

lstm <- embedding %>%
    layer_lstm(units = maxlen,dropout = 0.25, recurrent_dropout = 0.25, return_sequences = FALSE,
name = "lstm")

dense <- lstm %>%
    layer_dense(units = 128, activation = "relu", name = "dense")

predictions <- dense %>%
    layer_dense(units = 1, activation = "sigmoid", name = "predictions")
```

Model Architecture

```
model <- keras_model(input, predictions)                                      # Bring model together

# Freeze the embedding weights initially to prevent updates propagating back through and ruining our
embedding

get_layer(model, name = "embedding") %>%
  set_weights(list(fastwiki_embedding_matrix)) %>%
  freeze_weights()

# Compile

model %>% compile(
  optimizer = optimizer_adam(),
  loss = "binary_crossentropy",
  metrics = "binary_accuracy"
)

# Print architecture (plot_model isn't implemented in the R package yet)

print(model)
```

Model Architecture

```
Model
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input (InputLayer)           (None, None)              0
_________________________________________________________________
embedding (Embedding)        (None, None, 300)         4500000
_________________________________________________________________
lstm (LSTM)                  (None, 64)                93440
_________________________________________________________________
dense (Dense)                (None, 128)               8320
_________________________________________________________________
predictions (Dense)          (None, 1)                 129
=================================================================
Total params: 4,601,889
Trainable params: 101,889
Non-trainable params: 4,500,000
_________________________________________________________________
```

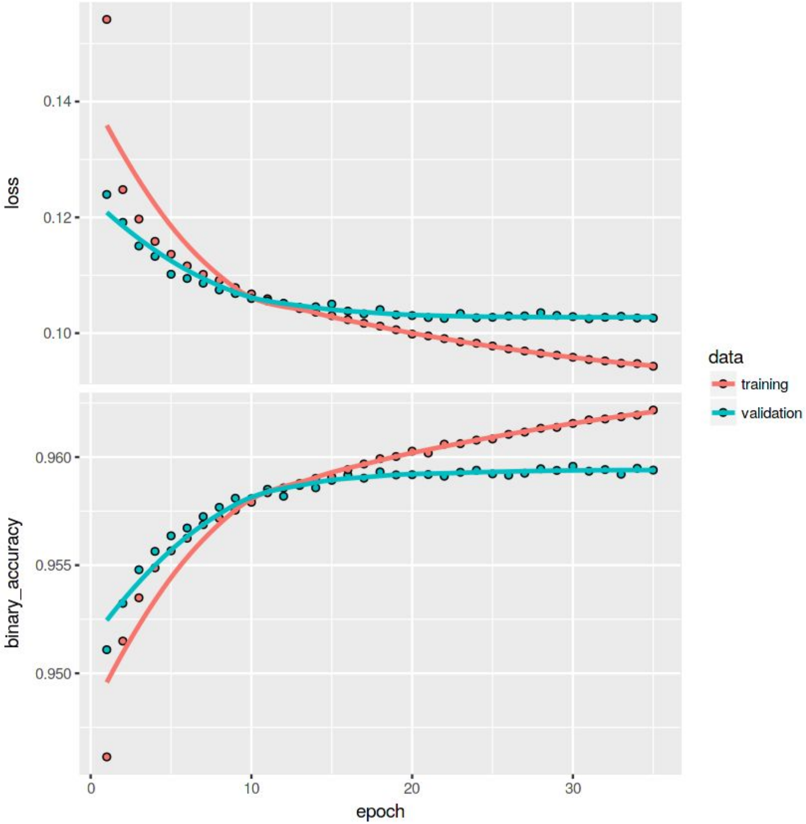

Tom Aindow https://www.kaggle.com/taindow

Model Training

```
# Train model

history <- model %>% fit(
  x_train,
  y_train,
  batch_size = 2048,
  validation_data = list(x_val, y_val),
  epochs = 35,
  view_metrics = FALSE,
  verbose = 0
)

# Look at training results

print(history)
plot(history)
```

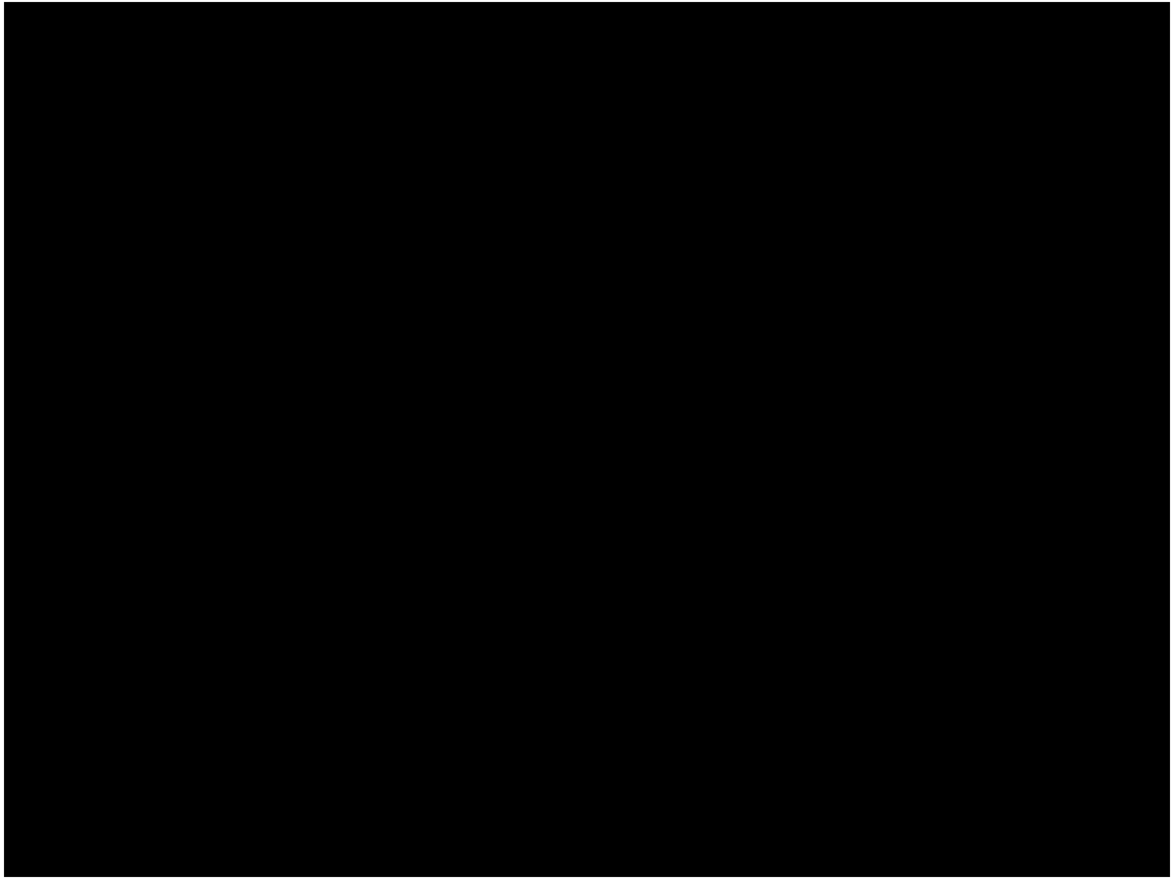
```
Trained on 1,175,509 samples, validated on
130,612 samples (batch_size=2,048, epochs=35)
Final epoch (plot to see history):
               loss: 0.09428
val_binary_accuracy: 0.9594
           val_loss: 0.1026
    binary_accuracy: 0.9622
```

Plots



Tom Aindow https://www.kaggle.com/taindow

A Metaphor




EXPLAIN YOURSELF
aMC HD
PREMIERE

LSTM Vis - Strobelt (et. all)

A **visual** analysis system for **exploring hypothesis** about **state dynamics** in Recurrent Neural Networks

LSTM Vis - Strobelt (et. all)

LSTMVis: A Tool for Visual Analysis of Hidden State Dynamics in Recurrent Neural Networks

Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M. Rush
– Harvard School of Engineering and Applied Sciences –

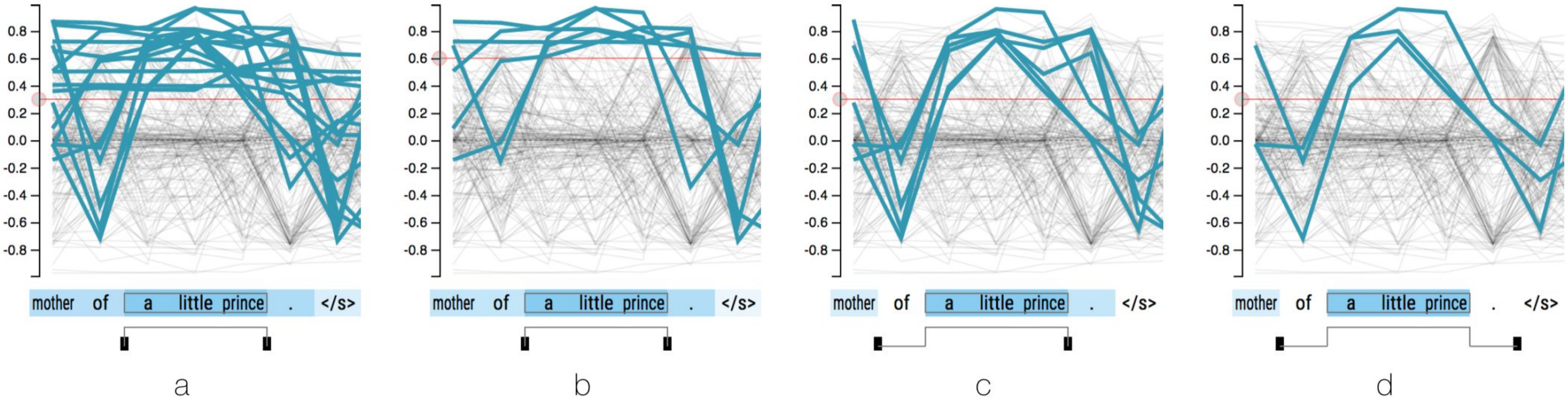
lstm.seas.harvard.edu/

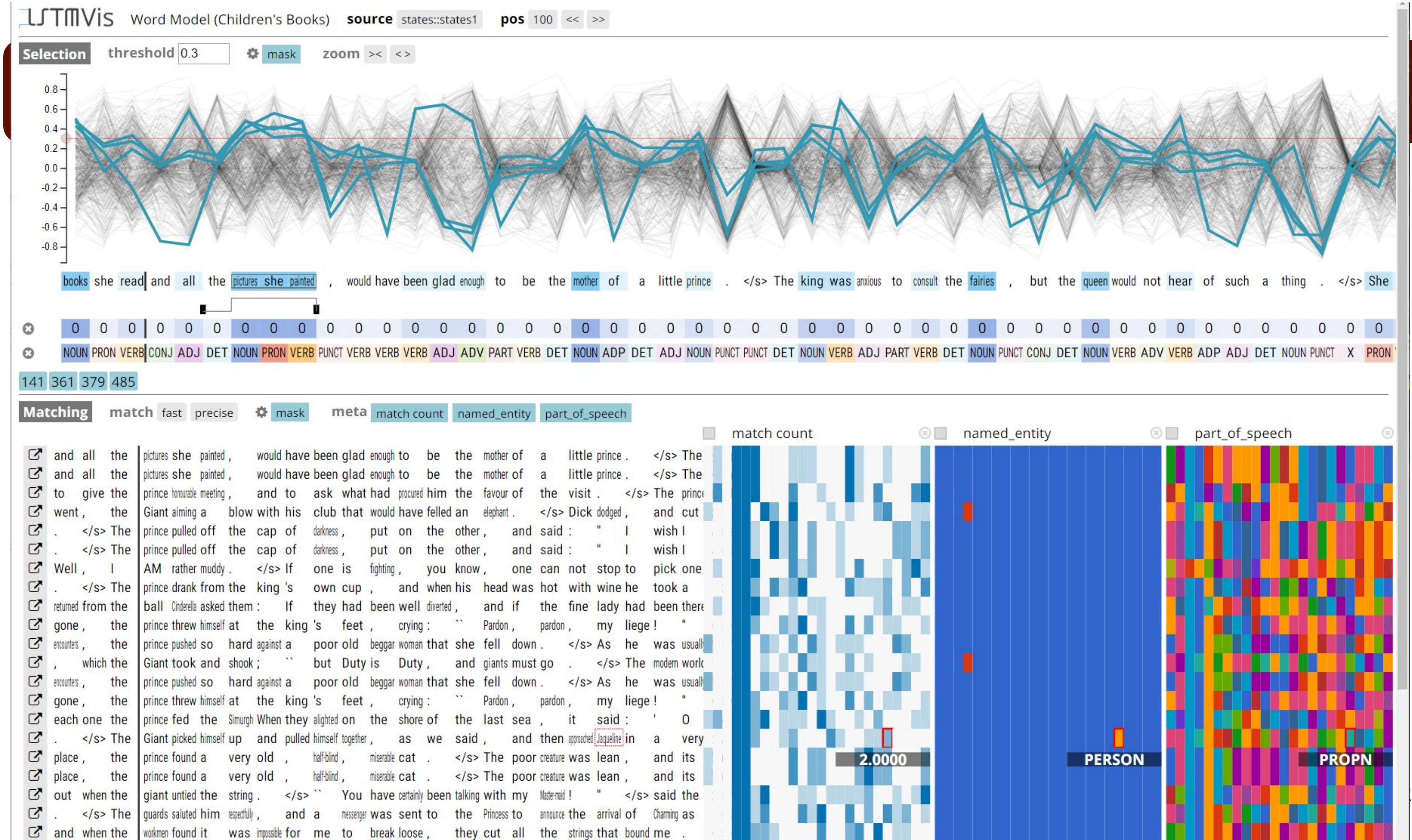

LSTMVis
Childbook with TopK
source states::states1
pos 115 << >>
Selection
threshold 0.3
mask
zoom >< <>
i1
i2
t
c
d
e1
e2
a
add meta track
37 149 190 197 269 278 320
Matching
match fast precise
mask stats
meta match count ner pos
match count
pos
7.0000
ADJ
g1
h
g2
b
f

A Trivial Language

```
( 1 ( 2 ) () ) 0 ( ( ( 3 ) ) 1 )
```


LSTMVis
Character Model (Parentheses)
source
states::states1
pos
100
<<
>>
Selection
threshold
0.3
mask
zoom
><
< >
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
) 2) ( ) (2 ( )) 1 ( ) ( ) ( ) (2 ( )) 1) (1 1) 0 (( ) 1) 0
339
Matching
match
fast
precise
mask
meta
match count
match count
) 1 (2 2 (3 3 3 ( ) (4) (4) ( )) 2 ( ) (
1 1 (2 2 (3 3 3 (4) ( ) (4))) 1 1 1 ((
2) (2 (3 (4) (4 4 4 4) 3) (3) ( )))
1 1 (2 (3 3 3 ( ) (4) ( ) (4) 3 (4)) (
2) (2 (3 (4 4) 3))) ( ) 0 0 0 0 0 ( ) (
( ) (2 (3)) (2 2 ((4) ( ) 3 (4)) (3
(1 (2 (3 3 (4))) ( ) (( ) 2) 1 1 1) 0
(1 (2 (3 3 (4 4) ( ) 3 3 3 (4)) (3))


0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
mother of a little prince . </s>
a
b
c
d


LSTMVis
Word Model (Children's Books)
source states::states1
pos 100 << >>
Selection
threshold 0.3
mask
zoom >< < >
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
books she read and all the pictures she painted , would have been glad enough to be the mother of a little prince . </s> The king was anxious to consult the fairies , but the queen would not hear of such a thing . </s> She
NOUN PRON VERB CONJ ADJ DET NOUN PRON VERB PUNCT VERB VERB VERB ADJ ADV PART VERB DET NOUN ADP DET ADJ NOUN PUNCT PUNCT DET NOUN VERB ADJ PART VERB DET NOUN PUNCT CONJ DET NOUN VERB ADV VERB ADP ADJ DET NOUN PUNCT X PRON
141 361 379 485
Matching
match fast precise
mask
meta match count named_entity part_of_speech
match count
named_entity
part_of_speech
2.0000
PERSON
PROPN

LSTMVis - Visual Analysis for Recurrent Neural Networks

LSTMVis allows you to interactively analyze the hidden state vectors of a recurrent neural network model, by simply selecting and comparing regions of the input. To demonstrate the system, we have provided a set of real example models and datasets to play with, including several word and character language models for text, music and code, a sequence auto-encoder, a German <-> English neural translation system, and a sentence summarization system. We recommend that you begin with the parentheses dataset, which has a simple and regular structure to demonstrate the use of the tool.

Character Model (Parentheses)

A simple synthetic dataset to test bracketing and counting. The language can open and close parents, and generate a number indicating the current nesting level.

meta: ---
index: false
length: 1000001

layer 1 cell (size: 999600 x 650)
layer 1 hidden (size: 999600 x 650)

Word Model (Children's Books)

A 1x200 LSTM language model trained on the Gutenberg Children's Book corpus.

meta: named_entity, part_of_speech
index: true
length: 1271912

layer 1 cell (size: 1271900 x 200)
layer 1 hidden (size: 1271900 x 200)

Word Model (Children's Books)

A 4x500 LSTM language model trained on the Gutenberg Children's Book corpus.

meta: named_entity, part_of_speech
index: true
length: 1271912

layer 1 cell (size: 317975 x 500)
layer 1 hidden (size: 317975 x 500)
layer 4 cell (size: 317975 x 500)
layer 4 hidden (size: 317975 x 500)

Word Model with Shallow Grammar

A 2x650 LSTM language model trained on the Wall Street Journal. Annotated with the CoNLL 2003 chunking and part-of-speech tag information.

meta: chunks, named_entities, tags
index: true
length: 221121

layer 1 cell (size: 221121 x 650)
layer 1 hidden (size: 221121 x 650)

Word Model (Wall Street Journal)

A 2x650 LSTM language model trained on the Wall Street Journal. Annotated with gold-standard part-of-speech tags.

meta: named_entity, part_of_speech
index: true
length: 929589

layer 1 cell (size: 928900 x 650)
layer 1 hidden (size: 928900 x 650)
layer 2 cell (size: 928900 x 650)

Character Model (Wall Street Journal)

A 2x650 LSTM character language model trained on the Wall Street Journal.

meta: vowels
index: false
length: 4879470

layer 1 cell (size: 840000 x 650)
layer 1 hidden (size: 840000 x 650)
layer 2 cell (size: 840000 x 650)
layer 2 hidden (size: 840000 x 650)

Sequence Auto-Encoder

The encoder states of an attention-based sequence auto-encoder. Trained on the Wall Street Journal. Implemented using seq2seq-attn.

meta: named_entity, part_of_speech
index: true
length: 100000

layer 2 cell (size: 100000 x 500)
layer 2 hidden (size: 100000 x 500)

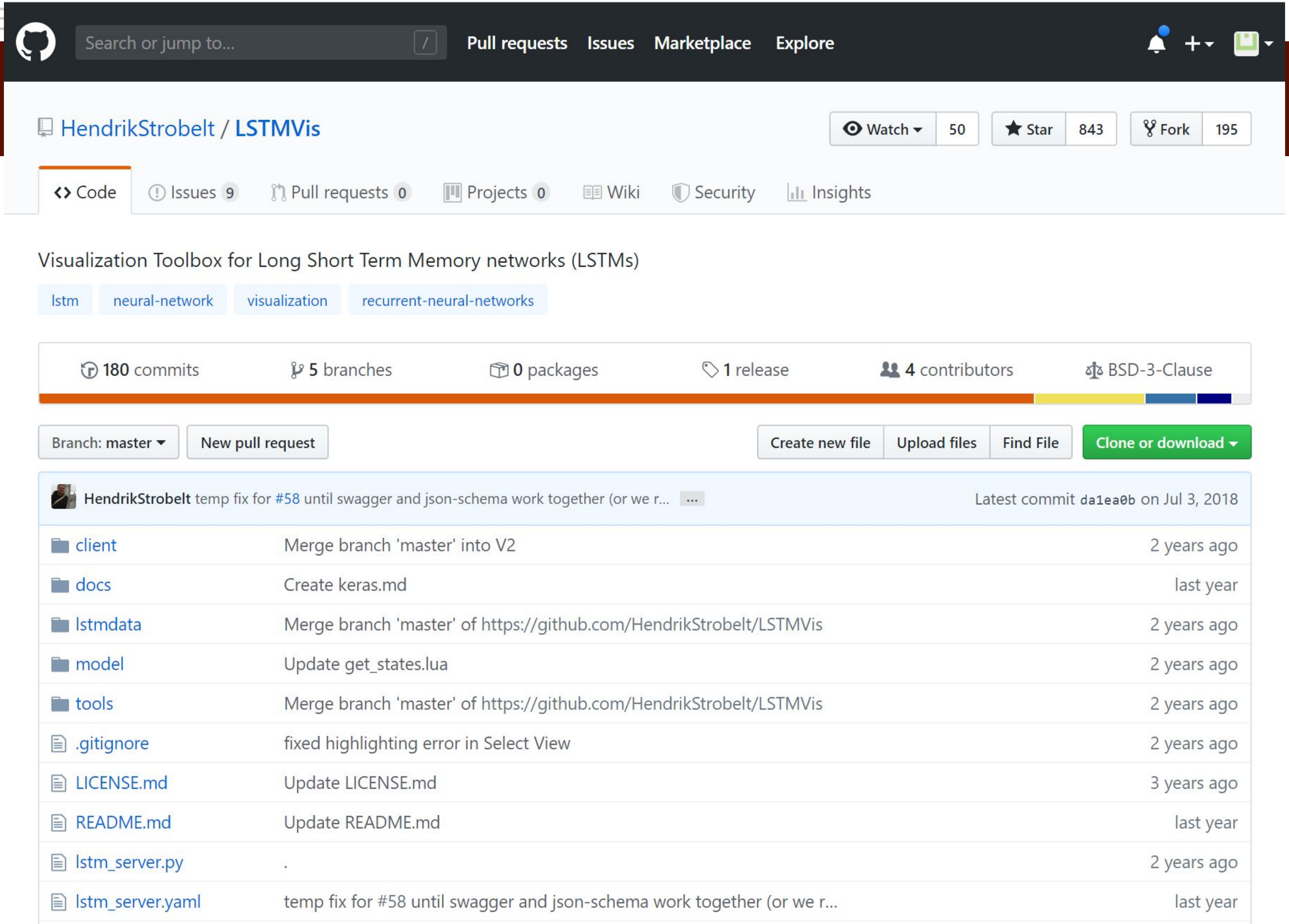
Word Model (Java)

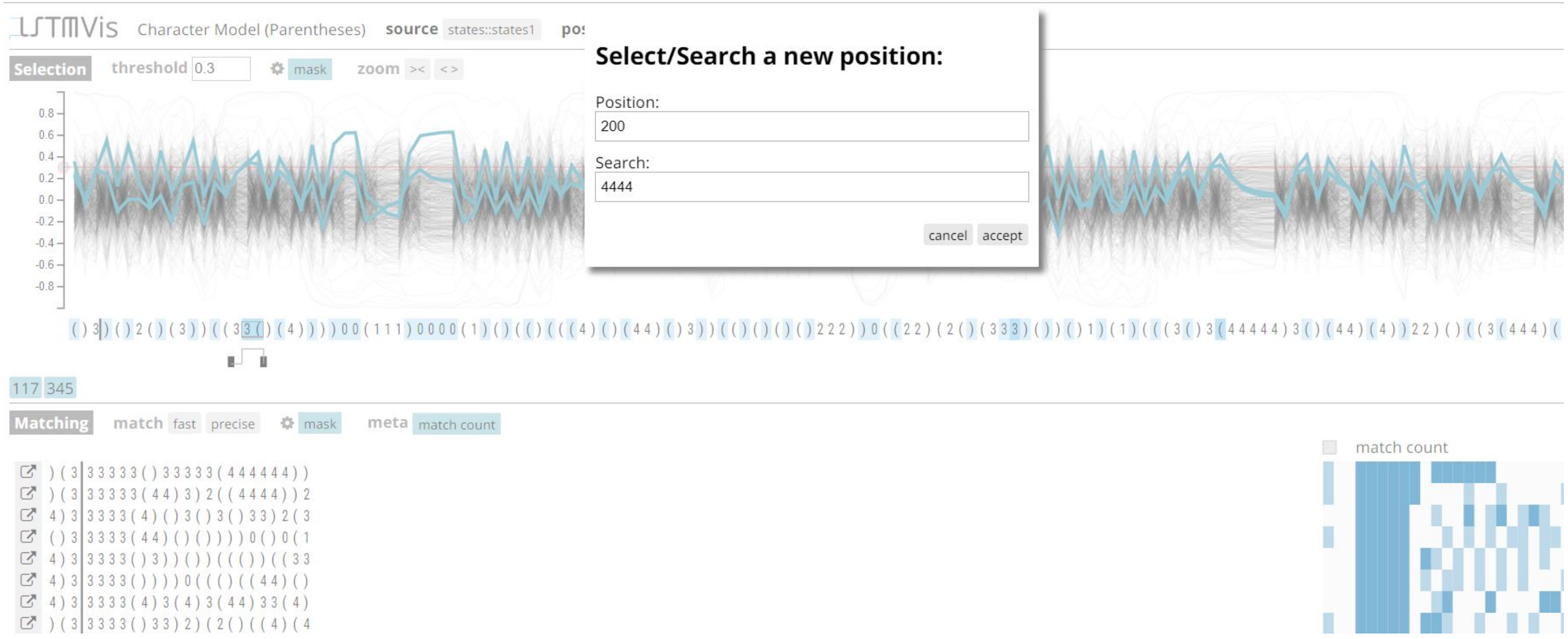
A 2x300 LSTM language model trained on tokenized Java code.

meta: lex
index: true
length: 1326608

layer 1 cell (size: 331625 x 300)
layer 1 hidden (size: 331625 x 300)
layer 2 cell (size: 331625 x 300)
layer 2 hidden (size: 331625 x 300)

Why E


Search or jump to...
Pull requests
Issues
Marketplace
Explore
HendrikStrobelt / LSTMVis
Watch 50
Star 843
Fork 195
Code
Issues 9
Pull requests 0
Projects 0
Wiki
Security
Insights
Visualization Toolbox for Long Short Term Memory networks (LSTMs)
lstm
neural-network
visualization
recurrent-neural-networks
180 commits
5 branches
0 packages
1 release
4 contributors
BSD-3-Clause
Branch: master
New pull request
Create new file
Upload files
Find File
Clone or download
HendrikStrobelt temp fix for #58 until swagger and json-schema work together (or we r...
Latest commit da1ea0b on Jul 3, 2018
client Merge branch 'master' into V2 2 years ago
docs Create keras.md last year
lstmdata Merge branch 'master' of https://github.com/HendrikStrobelt/LSTMVis 2 years ago
model Update get_states.lua 2 years ago
tools Merge branch 'master' of https://github.com/HendrikStrobelt/LSTMVis 2 years ago
.gitignore fixed highlighting error in Select View 2 years ago
LICENSE.md Update LICENSE.md 3 years ago
README.md Update README.md last year
lstm_server.py . 2 years ago
lstm_server.yaml temp fix for #58 until swagger and json-schema work together (or we r... last year


LSTMVis
Character Model (Parentheses)
source
states::states1
Selection
threshold
0.3
mask
zoom
Select/Search a new position:
Position:
200
Search:
4444
cancel
accept
117 345
Matching
match
fast
precise
mask
meta
match count
match count

Next Steps

- Profile training data
- Identify frequently used phrases
- Enable users to pass phrases into model, see results with LSTM Vis
- Deliver in a user friendly interface


HOORAY!!!


T. HANKS

UNUSED SLIDES

Seq2Seq Vis - Strobelt (et all)

SEQ2SEQ-VIS : A Visual Debugging Tool for Sequence-to-Sequence Models

Hendrik Strobelt*, Sebastian Gehrmann*, Michael Behrisch, Adam Perer, Hanspeter Pfister, Alexander M. Rush

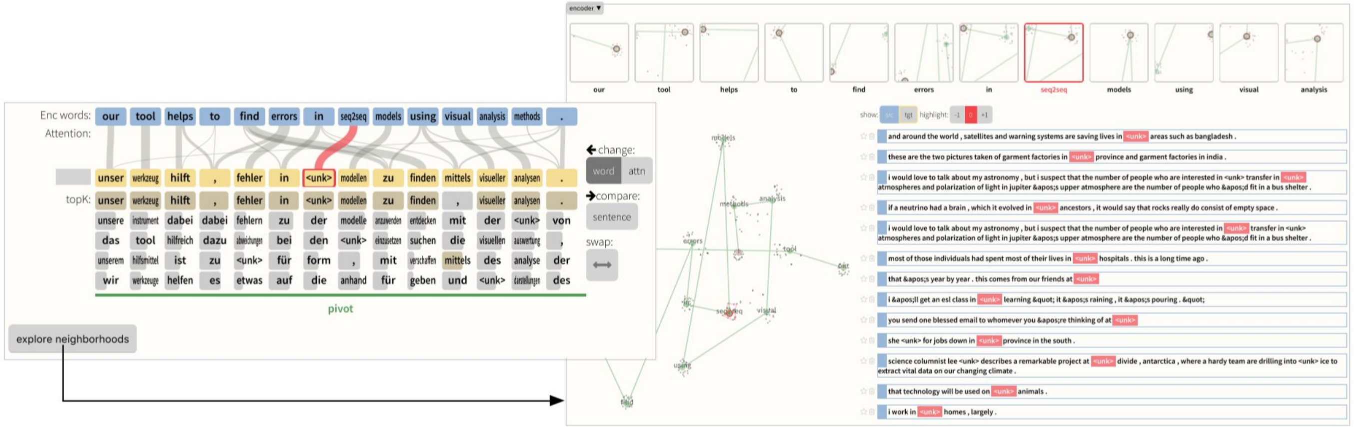


Fig. 1. Example of Seq2Seq-Vis. In the translation view (left), the source sequence *"our tool helps to find errors in seq2seq models using visual analysis methods."* is translated into a German sentence. The word *"seq2seq"* has correct attention between encoder and decoder (red highlight) but is not part of the language dictionary. When investigating the encoder neighborhoods (right), the user sees that *"seq2seq"* is close to other unknown words *"⟨unk⟩"*. The buttons enable user interactions for deeper analysis.


Me
Deep learning
Simple problem

Dense Word Vector (Sample)

lines [100]
[1] "work -0.0720 0.0130 -0.0777 -0.0491 -0.0850 -0.0337 -0.0243 0.0472 -0.0575 -0.0356 0.0931 -0.0579 -0.0667 -0.0612 -0.0634 -0.0520 0.1209 -0.0841 0.0455 0.0819 -0.0535 0.0750 0.0889 0.1769 0.0675 -0.0347 -0.0773 -0.0488 -0.0569 0.0227 0.0035 0.0395 0.0183 0.0542 -0.0188 0.0203 -0.0252 -0.0498 0.0278 0.0521 0.0046 -0.0992 0.0415 -0.0340 0.0507 -0.0419 -0.0607 0.0656 0.0076 0.0696 -0.0558 0.0456 -0.6356 -0.0703 -0.1587 -0.1104 0.0550 0.0160 0.0074 0.0279 -0.0508 0.0394 -0.0134 -0.0315 -0.0289 -0.0604 0.0277 0.0170 -0.0156 0.0355 0.0201 -0.0229 0.1151 -0.0961 0.0129 0.0488 0.0038 -0.0607 0.0469 0.1148 0.0382 0.0008 -0.0133 -0.1429 0.0423 -0.0841 -0.0146 -0.0523 0.0353 -0.0013 0.1078 0.0076 0.0498 -0.0432 -0.0121 -0.0540 0.0992 0.0559 -0.0699 0.0376 -0.1434 -0.0286 -0.0596 0.0298 0.0157 0.0019 0.0178 -0.1016 0.0129 -0.1058 -0.1313 -0.0973 0.0014 0.0206 -0.0798 -0.0549 0.0590 0.0583 -0.0561 -0.2958 0.0362 0.0613 0.0163 0.0457 -0.0181 0.1652 0.0427 0.0958 0.0169 -0.0537 -0.0391 -0.0483 -0.1888 -0.0876 -0.0107 0.1283 0.0102 0.0256 0.0357 0.1059 -0.0491 0.0576 0.0328 0.3096 0.0363 0.0040 0.0645 0.0361 -0.0414 0.0191 -0.0777 0.0496 0.0105 -0.0926 0.0151 0.0168 0.0571 -0.0834 0.1046 0.0154 -0.0041 0.1191 0.0743 0.0206 -0.0660 -0.0773 0.1190 -0.1354 0.0724 0.0236 0.1245 0.0727 0.0282 0.0034 -0.0738 -0.1089 0.1461 -0.2987 -0.0121 0.1115 -0.0153 0.0245 -0.0143 -0.0014 0.0816 -0.1982 -0.0084 -0.0841 -0.0579 -0.0007 0.0116 0.0352 -0.0174 -0.0747 0.0576 0.0116 0.0162 0.0358 0.1240 -0.0799 0.0493 -0.1226 0.0315 -0.0136 0.0567 0.0402 -0.0510 -0.1615 0.0544 0.0628 0.1056 -0.0529 -0.0183 -0.0143 -0.0088 0.0167 -0.0576 -0.0219 0.0124 -0.0323 -0.1009 -0.0071 0.1330 0.0761 -0.0211 -0.0928 -0.1289 0.0723 -0.0882 0.0380 -0.1981 0.0467 0.3405 -0.0776 -0.0254 -0.1370 -0.0185 0.0508 -0.1630 0.0238 -0.0336 -0.1350 -0.0180 -0.0343 0.1136 0.0755 -0.0034 -0.0069 0.0272 0.3542 0.0856 0.0476 -0.1113 0.0011 -0.0516 0.0652 0.0673 0.0236 0.0239 -0.0063 -0.1020 -0.0650 -0.0401 0.1212 -0.1891 -0.0666 0.0407 -0.0919 -0.0664 0.0010 -0.0024 0.0990 -0.0738 -0.0545 0.0480 0.0535 -0.1183 0.0513 -0.0019 -0.0418 -0.0745 0.1153 0.1107 0.0297 0.0406 -0.0207 -0.0360 0.1072 0.0884 -0.0229 -0.0926 -0.0084 -0.0455 0.0076 -0.1298 0.0394 -0.0261 0.2044 -0.1034 -0.0521"