TeMS: 17 Lines of Code That Got Me My Dream Job

Dalya Gartzman

TeMS: 17 Lines of Code That Got Me My Dream Job

Dalya Gartzman

TeMS -
Textual Math Solver

Simplisico

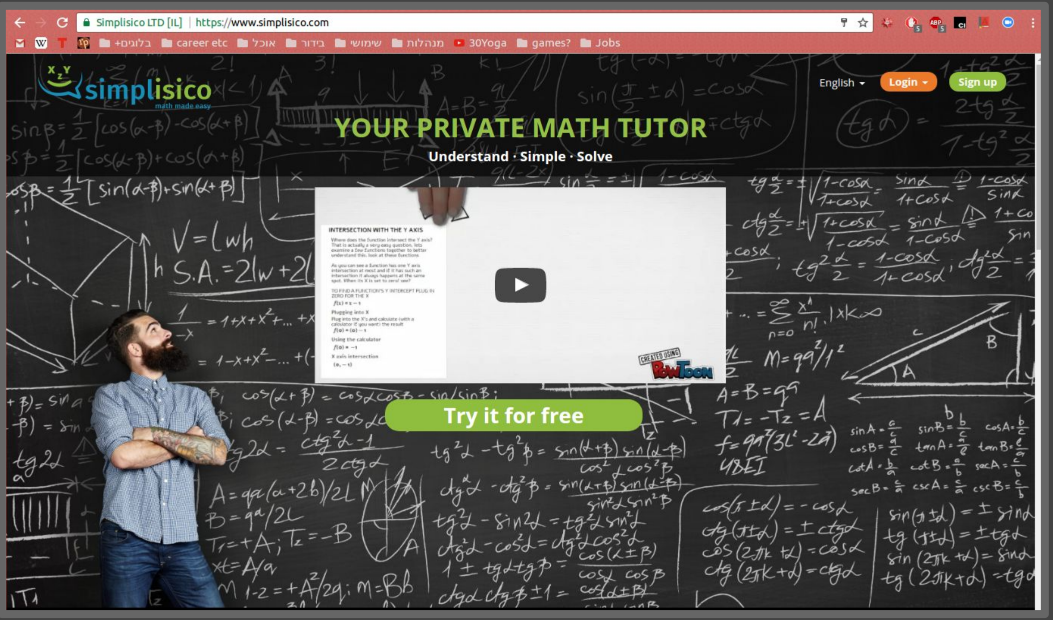
seematics

VISIONARY
DEEP LEARNING

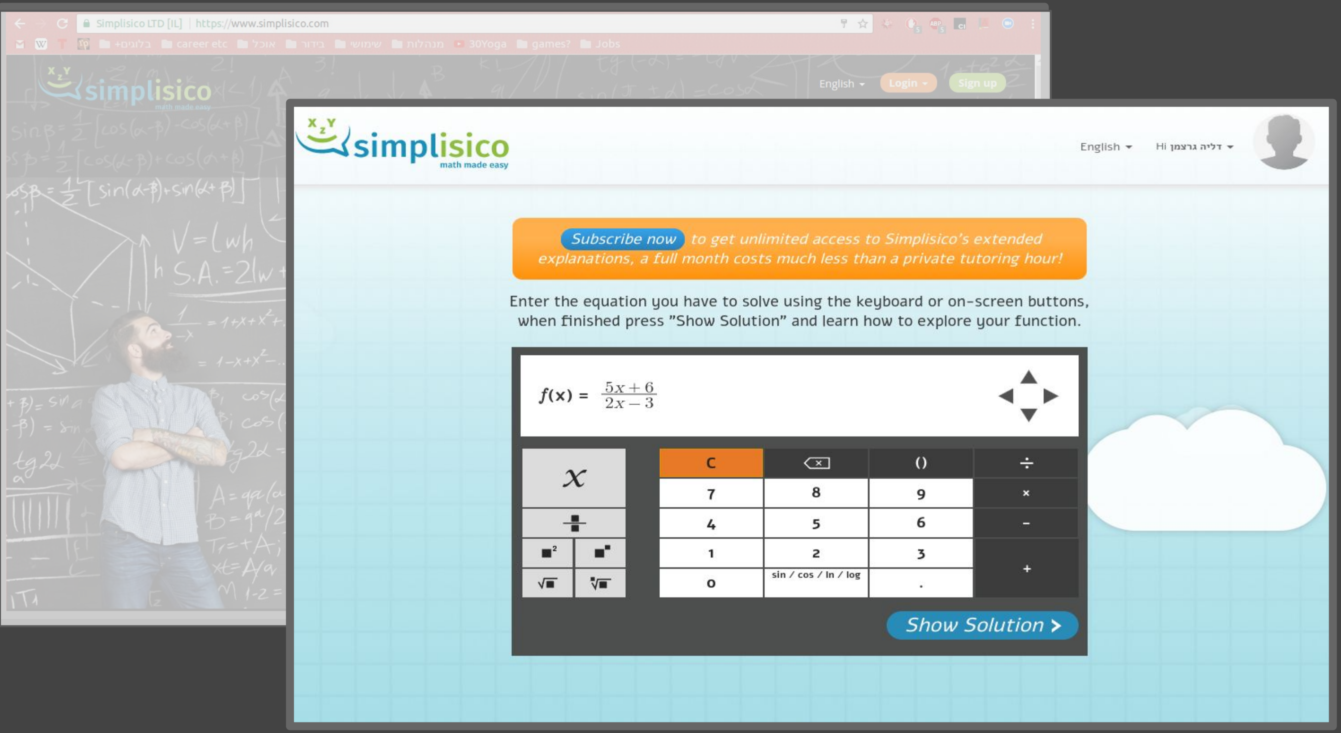
via

PART I - What is TeMS?

PART I - What is TeMS?



PART I - What is TeMS?

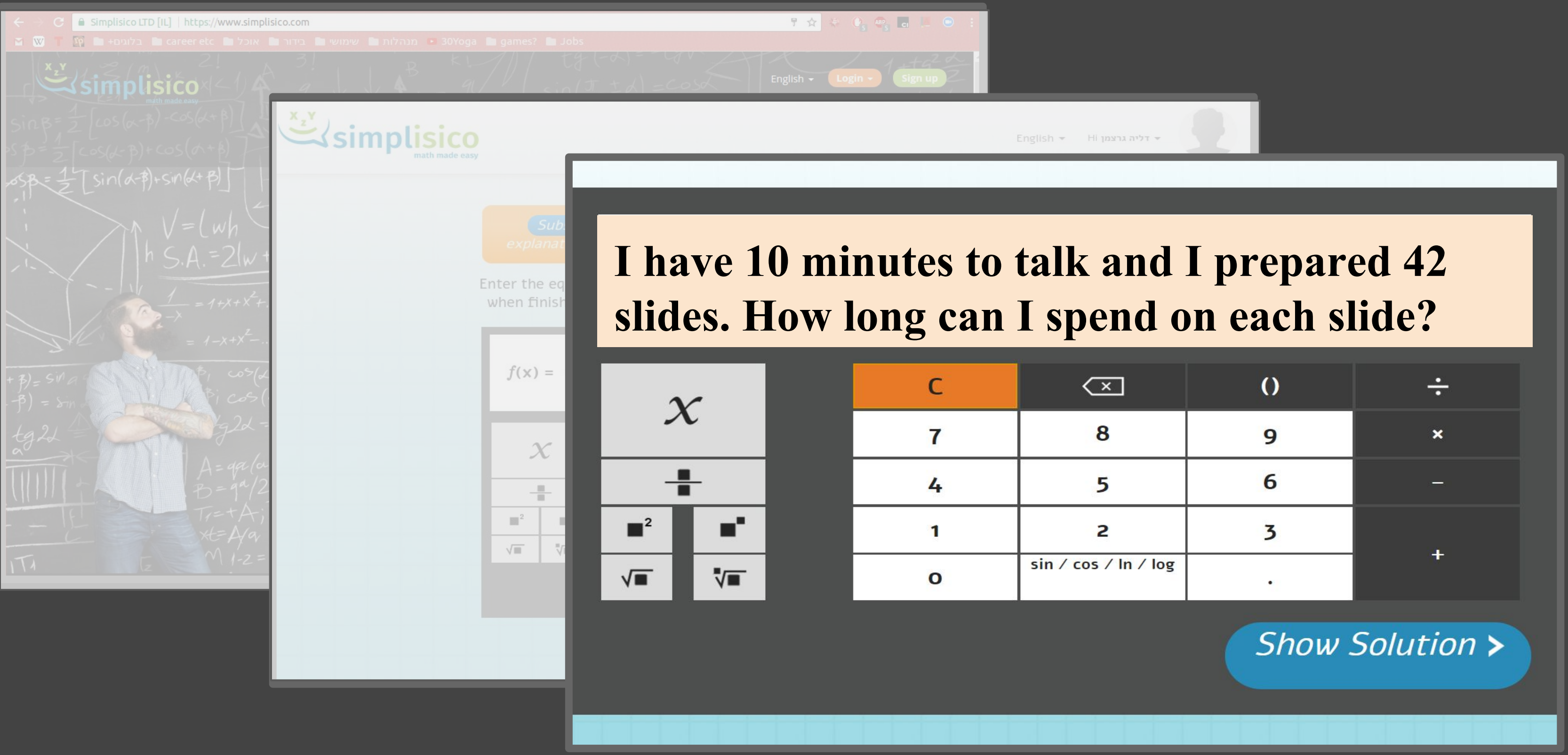


Subscribe now *to get unlimited access to Simplisico's extended explanations, a full month costs much less than a private tutoring hour!*

Enter the equation you have to solve using the keyboard or on-screen buttons, when finished press "Show Solution" and learn how to explore your function.

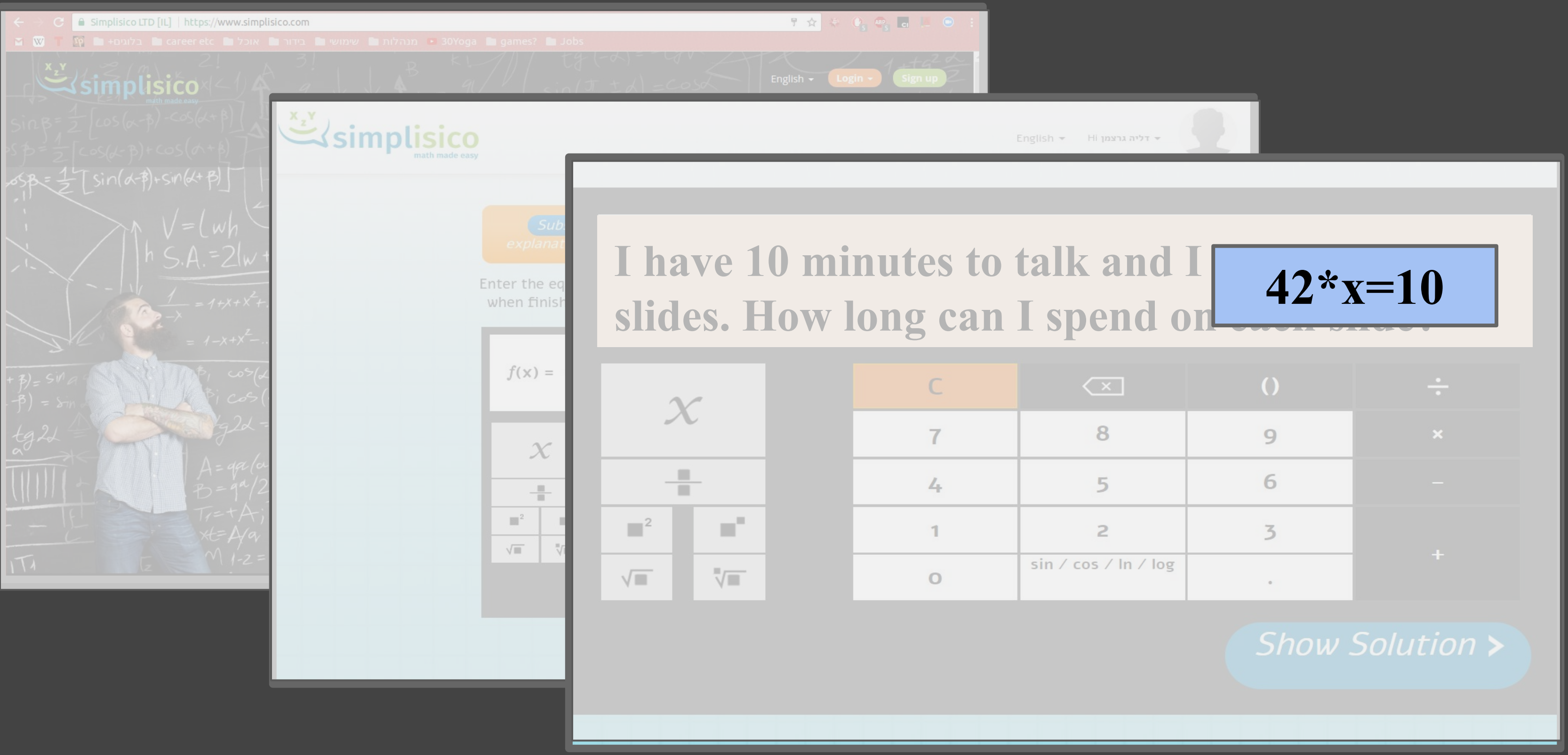
$f(x) = \frac{5x+6}{2x-3}$

PART I - What is TeMS?



I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?

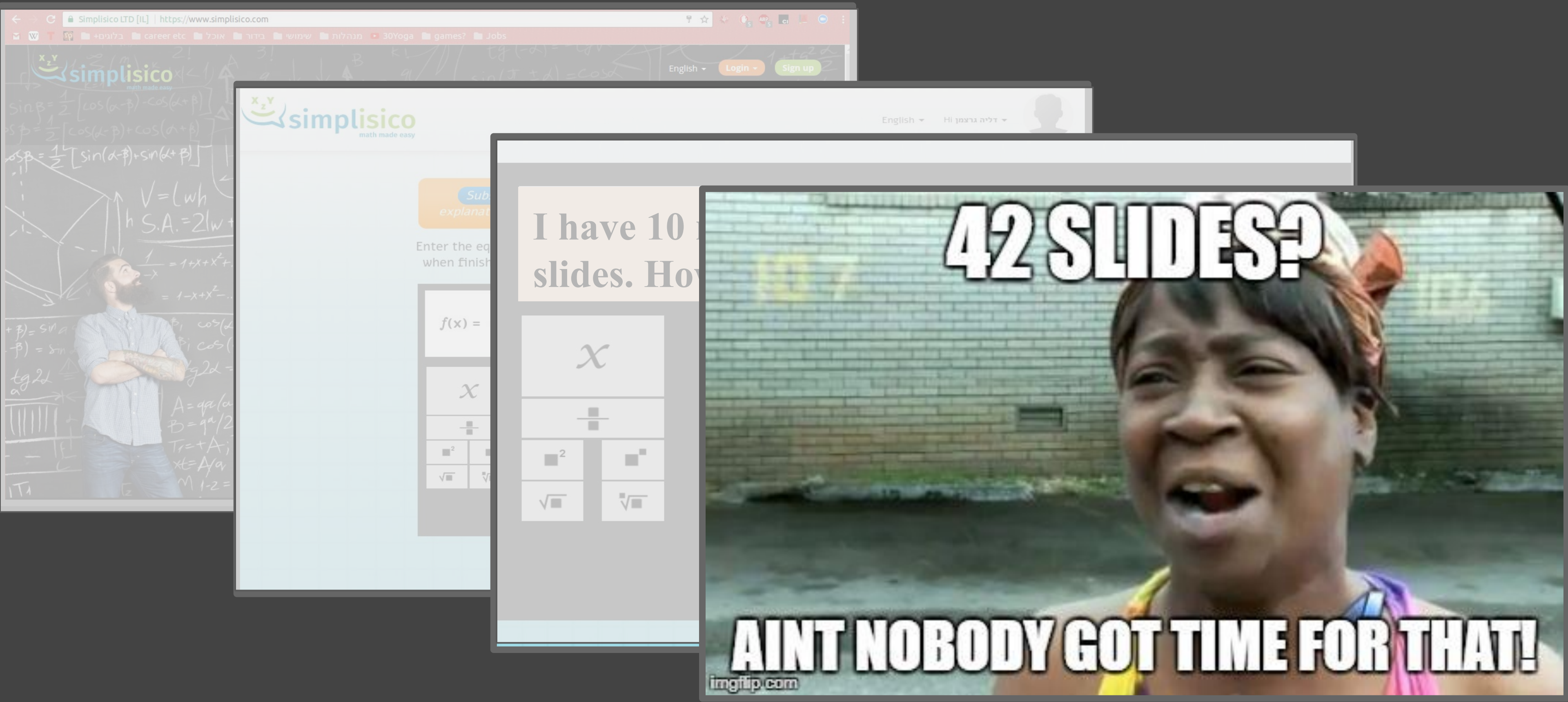
PART I - What is TeMS?



I have 10 minutes to talk and I
slides. How long can I spend on

42*x=10

PART I - What is TeMS?



(What is machine learning?)

(What is machine learning?)

human learning:

shape(●) = ?

(What is machine learning?)

human learning:

```
If (# ┌ ) = 0
    return circle
If (# ┌ ) = 4
    return square
```

shape(●) = ?

(What is machine learning?)

human learning:

```
If (# ┌) = 0
    return circle
If (# ┌) = 4
    return square
```

shape(●) = ?

machine learning:

shape(●) = ?

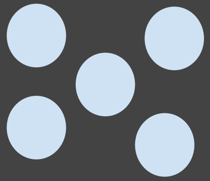
(What is machine learning?)

human learning:

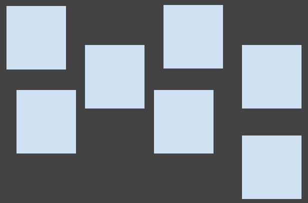
```
If (# ┌ ) = 0
    return circle
If (# ┌ ) = 4
    return square
```

shape(●) = ?

machine learning:



= circle



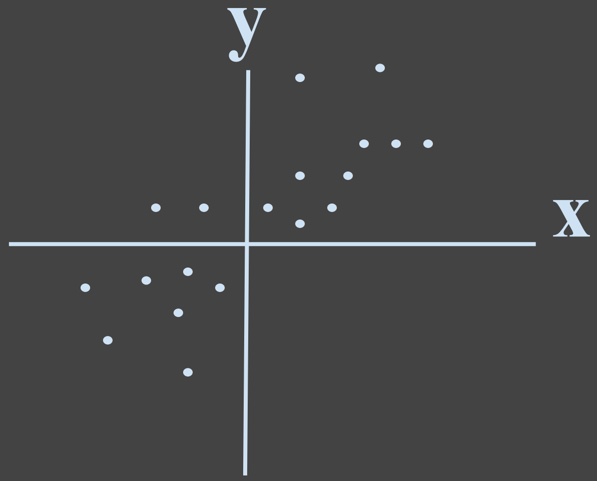
= square

shape(●) = ?

(What is a neaural network?)

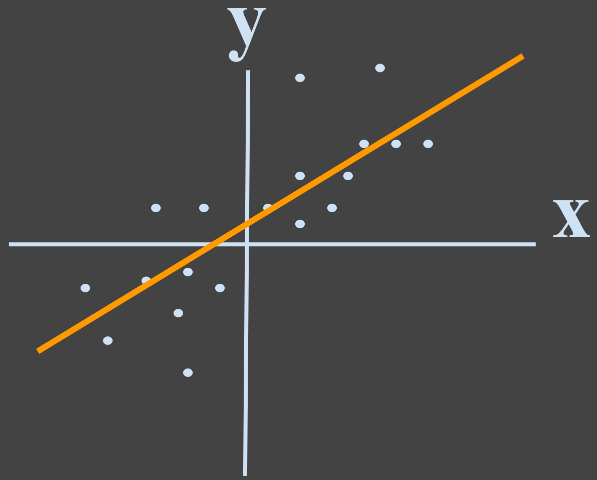
(What is a neaural network?)

neuron:



(What is a neaural network?)

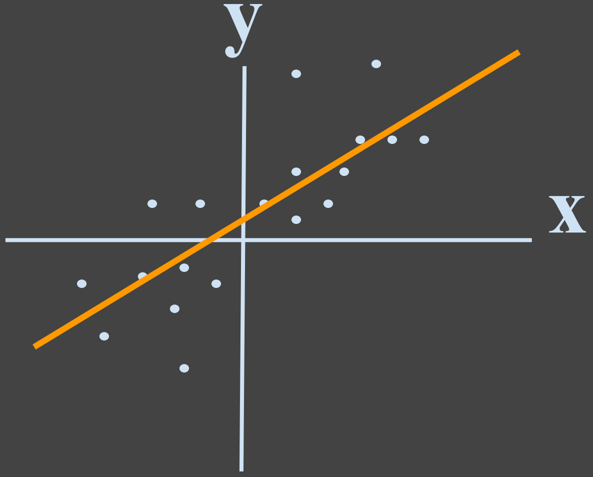
neuron:



y=f(x)

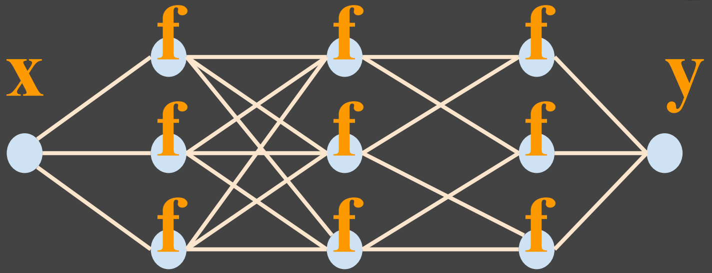
(What is a neaural network?)

neuron:



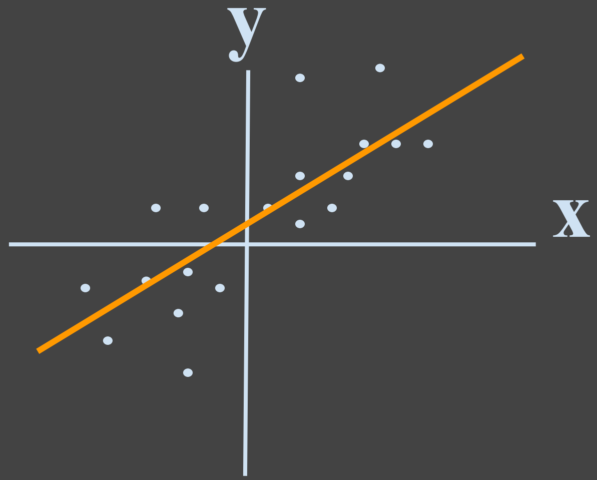
$y=f(x)$

neural network:



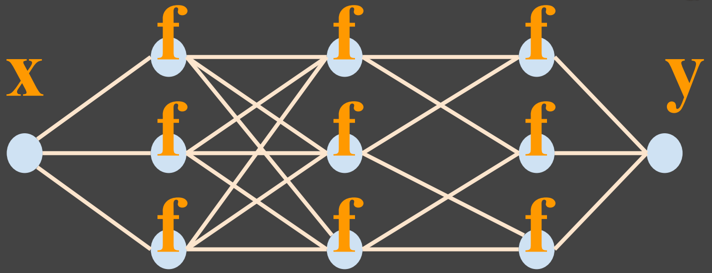
(What is a neaural network?)

neuron:



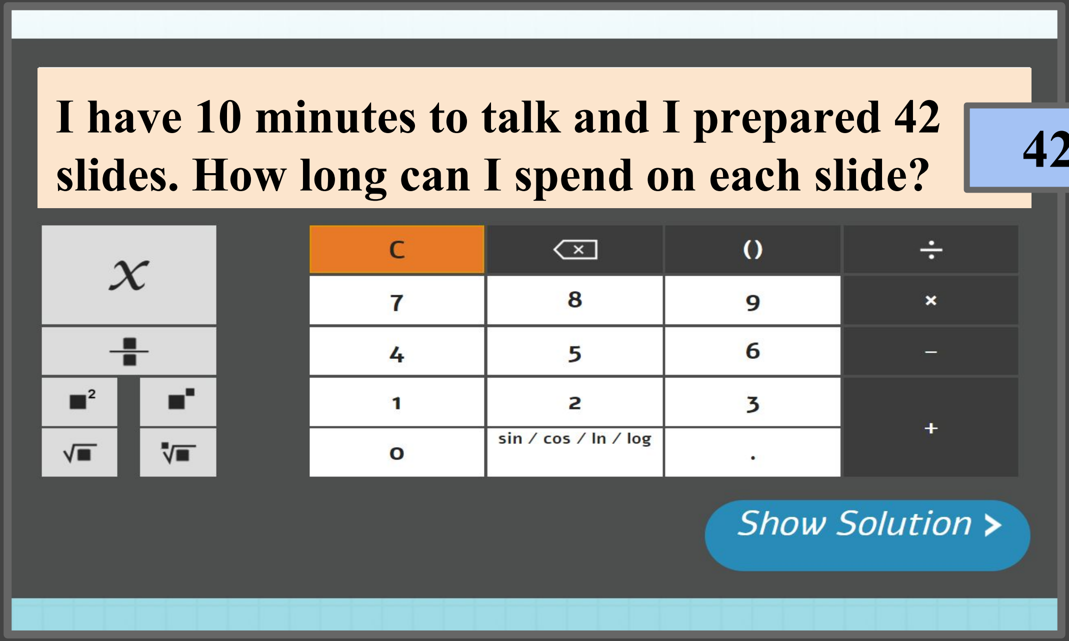
y=f(x)

neural network:



- architechture
- what are “f ”?
- more stuff...

PART I - What is TeMS?



I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?

$42*x=10$

Preprocessing

Preprocessing

I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?

Preprocessing

I have $\boxed{A}$ minutes to talk and I prepared $\boxed{B}$ slides. How long can I spend on each slide?

Preprocessing

I have A minutes to talk and I prepared B slides. How long can I spend on each slide?

TOKENIZE

[10, 42, 0] +

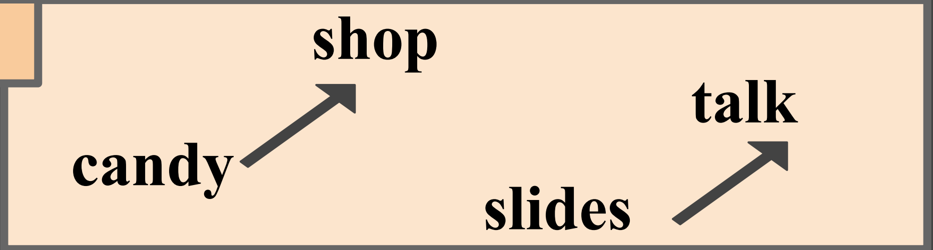
[... ‘have’, **‘varA’**, ‘minutes’, …
‘prepared’, **‘varB’**, ‘slides’, …]

Preprocessing

I have A minutes to talk and I prepared B slides. How long can I spend on each slide?

TOKENIZE [10, 42, 0] +
[... ‘have’, **‘varA’**, ‘minutes’, …
‘prepared’, **‘varB’**, ‘slides’, …]

EMBED



candy

shop

slides

talk

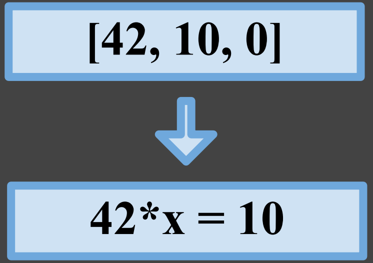
TeMS: a Sequence-to-Permutation RNN with LSTM and Attention

TeMS: a Sequence-to-Permutation RNN with LSTM and Attention

I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?

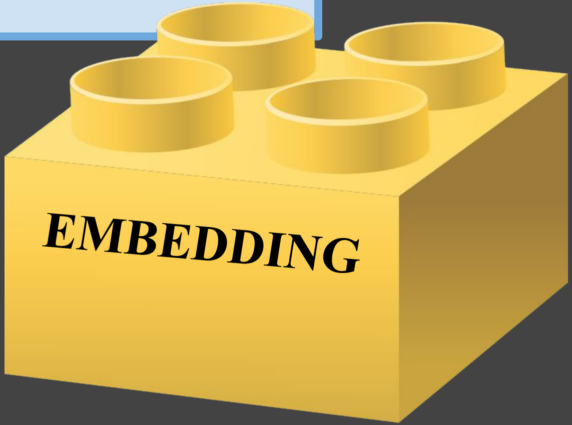
TeMS: a Sequence-to-Permutation RNN with LSTM and Attention

I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?



TeMS: a Sequence-to-Permutation RNN with LSTM and Attention

I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?

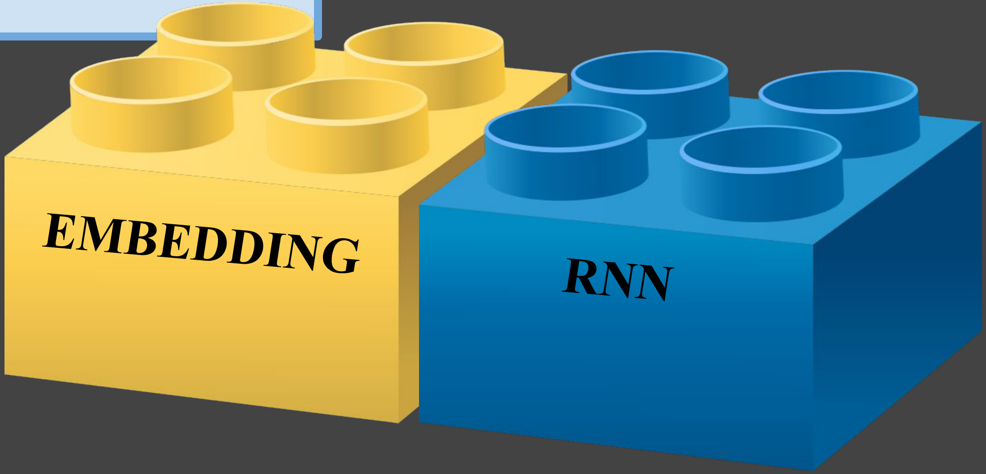


[42, 10, 0]

42*x = 10

TeMS: a Sequence-to-Permutation RNN with LSTM and Attention

I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?



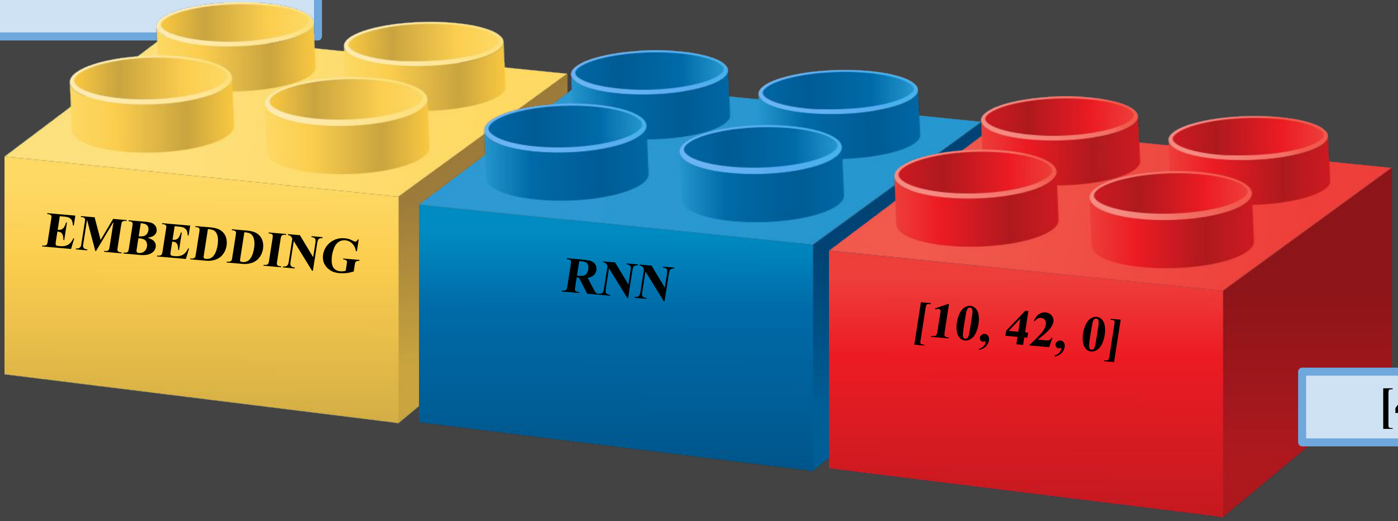
[42, 10, 0]

↓

42*x = 10

TeMS: a Sequence-to-Permutation RNN with LSTM and Attention

I have 10 minutes to talk and I prepared 42 slides. How long can I spend on each slide?

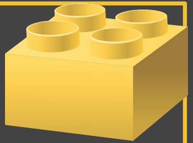


[42, 10, 0]

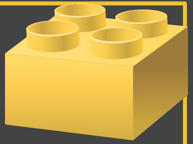
42*x = 10

```python
# generate embeddings
embedding = w2v.Word2Vec()
embedding.build_vocab(all_questions)
embedding.train(all_questions)

# Words model
words_model = Sequential()
words_model.add(Embedding(vocab_size, embedding_dim, weights=[embedding]))
words_model.add(LSTM(64, activation='softmax', return_sequences=True, dropout=0.5))
words_model.add(LSTM(64, activation='relu', return_sequences=False, dropout=0.5))
words_input = Input(shape=(max_len,))
processed_words = words_model(words_input)

# Numbers model
nums_model = Sequential()
nums_model.add(Dense(3, input_dim=3))
nums_input = Input(shape=(3,))
processed_nums = nums_model(nums_input)

# Concatenate models
merged = keras.layers.concatenate([processed_nums, processed_words])

# add one dense layer to integrate the merging
hidden = Dense(32, activation='tanh')(merged)
hidden = Dropout(0.5)(hidden)
# finish with a dense layer
output = Dense(3)(hidden)
```

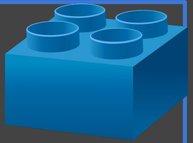
```
# generate embeddings
embedding = w2v.Word2Vec()
embedding.build_vocab(all_questions)
embedding.train(all_questions)
```



```
# Words model
words_model = Sequential()
words_model.add(Embedding(vocab_size, embedding_dim, weights=[embedding]))
words_model.add(LSTM(64, activation='softmax', return_sequences=True, dropout=0.5))
words_model.add(LSTM(64, activation='relu', return_sequences=False, dropout=0.5))
words_input = Input(shape=(max_len,))
processed_words = words_model(words_input)

# Numbers model
nums_model = Sequential()
nums_model.add(Dense(3, input_dim=3))
nums_input = Input(shape=(3,))
processed_nums = nums_model(nums_input)

# Concatenate models
merged = keras.layers.concatenate([processed_nums, processed_words])

# add one dense layer to integrate the merging
hidden = Dense(32, activation='tanh')(merged)
hidden = Dropout(0.5)(hidden)
# finish with a dense layer
output = Dense(3)(hidden)
```



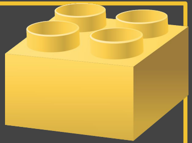
```
# generate embeddings
embedding = w2v.Word2Vec()
embedding.build_vocab(all_questions)
embedding.train(all_questions)
```



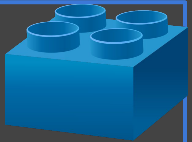
```
# Words model
words_model = Sequential()
words_model.add(Embedding(vocab_size, embedding_dim, weights=[embedding]))
words_model.add(LSTM(64, activation='softmax', return_sequences=True, dropout=0.5))
words_model.add(LSTM(64, activation='relu', return_sequences=False, dropout=0.5))
words_input = Input(shape=(max_len,))
processed_words = words_model(words_input)
```

```
# Numbers model
nums_model = Sequential()
nums_model.add(Dense(3, input_dim=3))
nums_input = Input(shape=(3,))
processed_nums = nums_model(nums_input)

# Concatenate models
merged = keras.layers.concatenate([processed_nums, processed_words])

# add one dense layer to integrate the merging
hidden = Dense(32, activation='tanh')(merged)
hidden = Dropout(0.5)(hidden)
# finish with a dense layer
output = Dense(3)(hidden)
```

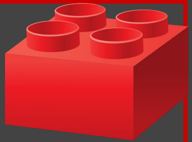
```
# generate embeddings
embedding = w2v.Word2Vec()
embedding.build_vocab(all_questions)
embedding.train(all_questions)
```



```
# Words model
words_model = Sequential()
words_model.add(Embedding(vocab_size, embedding_dim, weights=[embedding]))
words_model.add(LSTM(64, activation='softmax', return_sequences=True, dropout=0.5))
words_model.add(LSTM(64, activation='relu', return_sequences=False, dropout=0.5))
words_input = Input(shape=(max_len,))
processed_words = words_model(words_input)
```

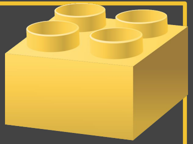


```
# Numbers model
nums_model = Sequential()
nums_model.add(Dense(3, input_dim=3))
nums_input = Input(shape=(3,))
processed_nums = nums_model(nums_input)

# Concatenate models
merged = keras.layers.concatenate([processed_nums, processed_words])
```

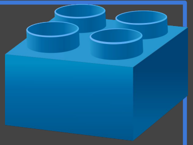


```
# add one dense layer to integrate the merging
hidden = Dense(32, activation='tanh')(merged)
hidden = Dropout(0.5)(hidden)
# finish with a dense layer
output = Dense(3)(hidden)
```

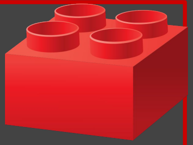
```
# generate embeddings
embedding = w2v.Word2Vec()
embedding.build_vocab(all_questions)
embedding.train(all_questions)
```



```
# Words model
words_model = Sequential()
words_model.add(Embedding(vocab_size, embedding_dim, weights=[embedding]))
words_model.add(LSTM(64, activation='softmax', return_sequences=True, dropout=0.5))
words_model.add(LSTM(64, activation='relu', return_sequences=False, dropout=0.5))
words_input = Input(shape=(max_len,))
processed_words = words_model(words_input)
```



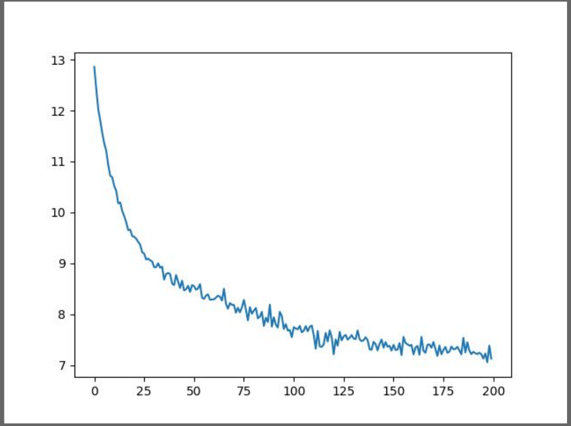
```
# Numbers model
nums_model = Sequential()
nums_model.add(Dense(3, input_dim=3))
nums_input = Input(shape=(3,))
processed_nums = nums_model(nums_input)

# Concatenate models
merged = keras.layers.concatenate([processed_nums, processed_words])
```



```
# add one dense layer to integrate the merging
hidden = Dense(32, activation='tanh')(merged)
hidden = Dropout(0.5)(hidden)
# finish with a dense layer
output = Dense(3)(hidden)
```

Results

Input:	Prediction:	Truth:
[‘You have 5 apples…’,	5*x=39-19	5*x=39-19
‘23 years from now…’,	2*x=23-16.8	2*x=23-16.8
‘Sean collected 47 stamps…’,	5*x=47-17	5*x=47-17
‘A cup of coffee costs 13.5$ ...’,	2*x=13.5-3.5	2*x=13.5-3.5
‘Fourteen drinks were served …’]	4*x=14-6	-4*x=6-14



PART I Recap

PART II - Get a Side Project!

PART II - Get a Side Project!

PART II - Get a Side Project!

“hello world”

PART II - Get a Side Project!

yes!

“hello world”

PART II - Get a Side Project!

“hello world”

yes!

publish???

PART II - Get a Side Project!

“hello world”

yes!

publish???

Take Home Message

Thank you :)

Questions?

DalyaG@gmail.com

TeMS -
Textual Math Solver

Simplisico

seematics

VISIONARY
DEEP LEARNING

via