Multi-Class Sentiment Classification for Short Text Sequences

TIMOTHY LIU KAIHUI

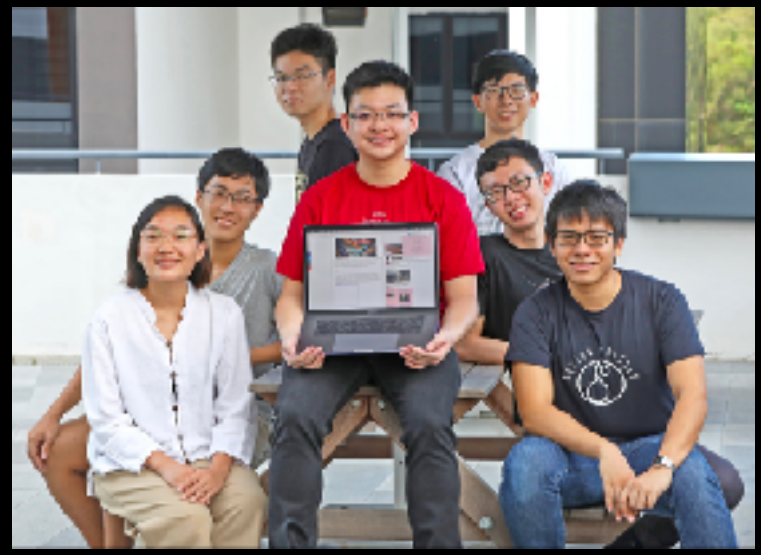
SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

What a selfless and courageous hero ... Willing to give his life for a total stranger .... Deepest sympathies and sincere condolences for his family and all in France

Sack the whole f***ing cabinet

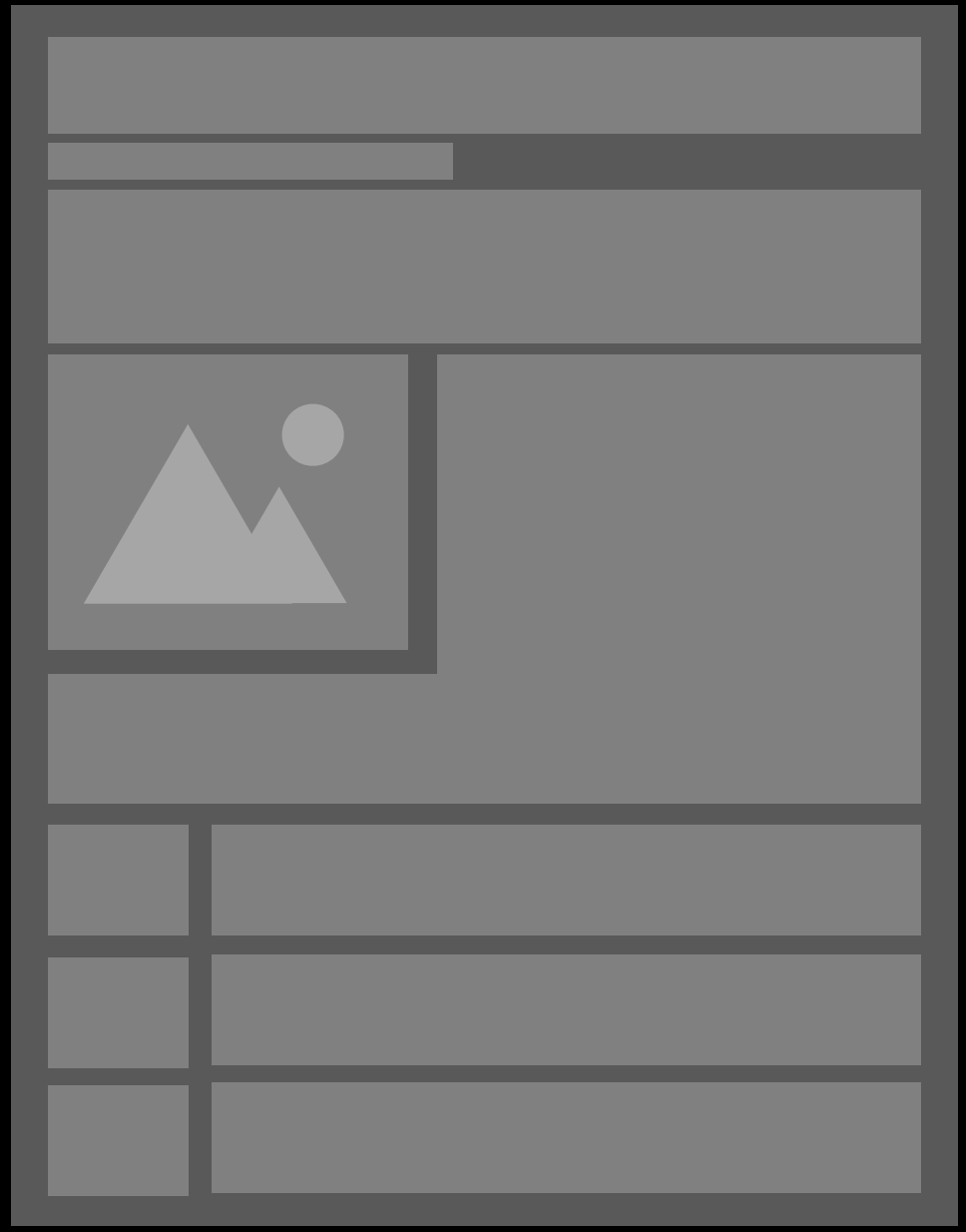
Motivation

- As part of a greater research project to tackle ’Fake News’
  - Team of students in SUTD



Motivation

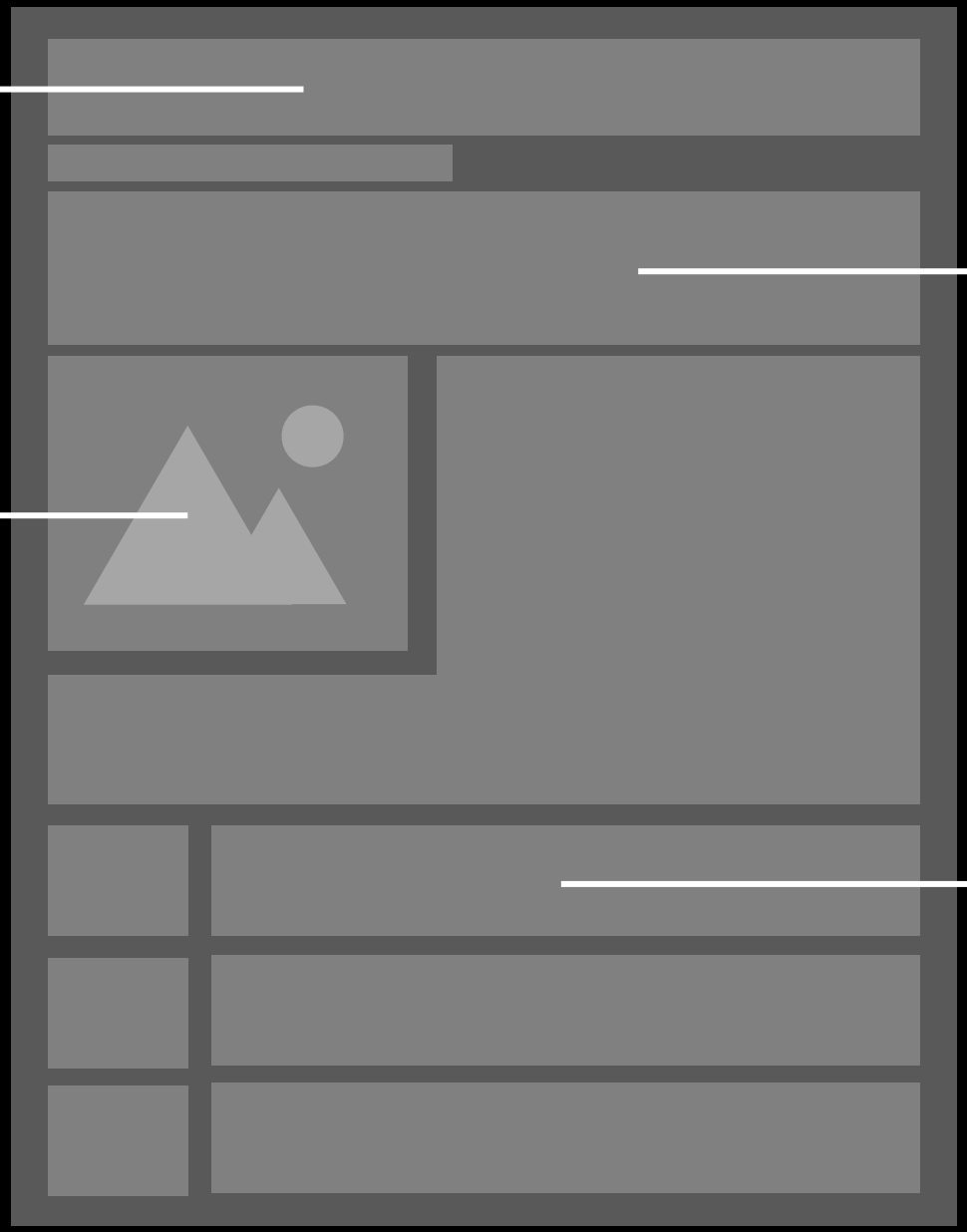
- Evaluate people’s reaction to online content
  - Record it as a “feature"
- Use this feature (among others) to flag suspicious content



Clickbait heading?

Quality of text?

Known hoax?



Audience Reaction

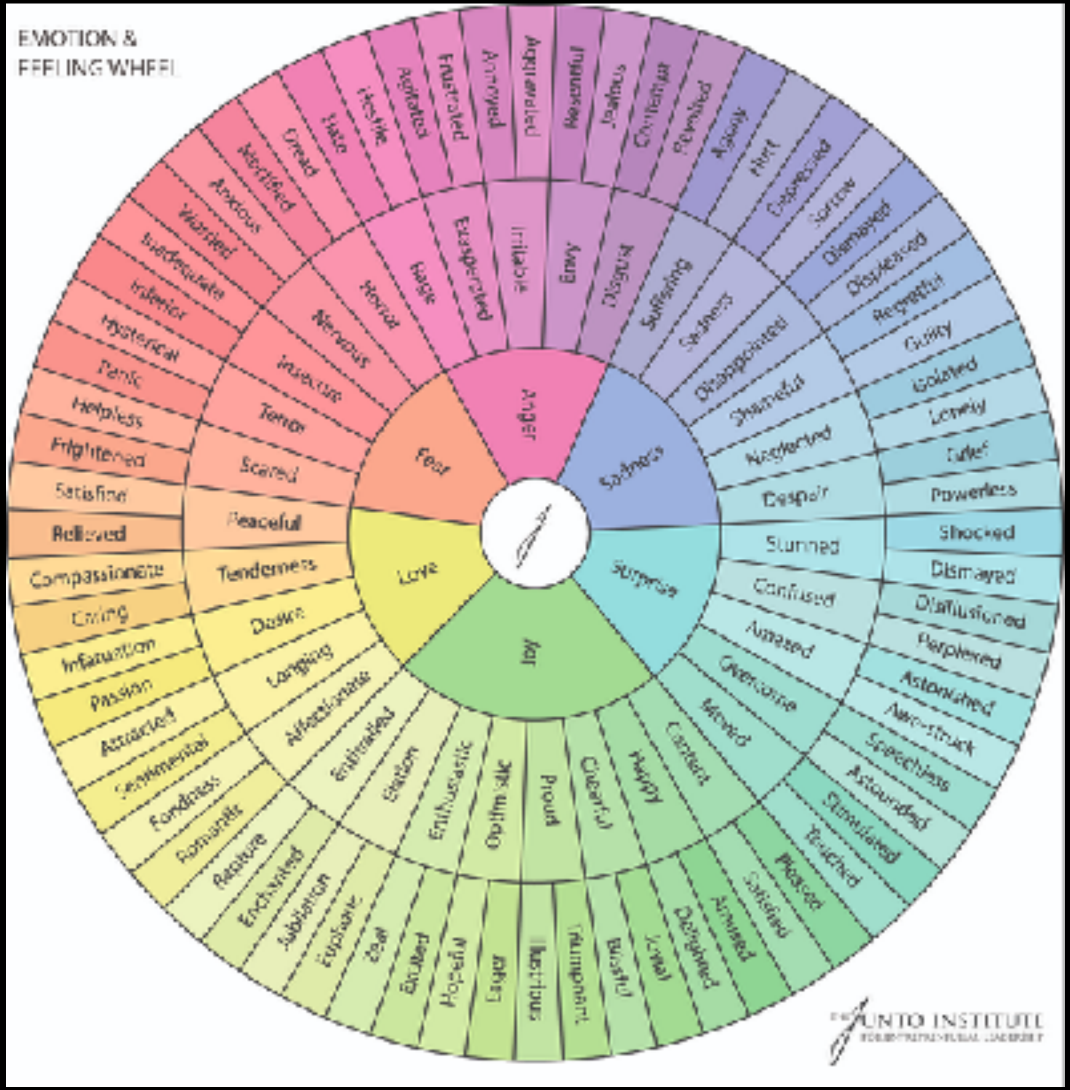
Challenges

- Appropriate Data
  - short–mid length, highly informal sentences
    - Expressing wide range of emotions
  - ~~IMDB Movie Reviews~~, ~~Amazon product reviews~~
  - Tweets?
- Label granularity (!!)
  - Positive/Negative is **not** enough to gauge reaction
  - Hard to find such labelled data

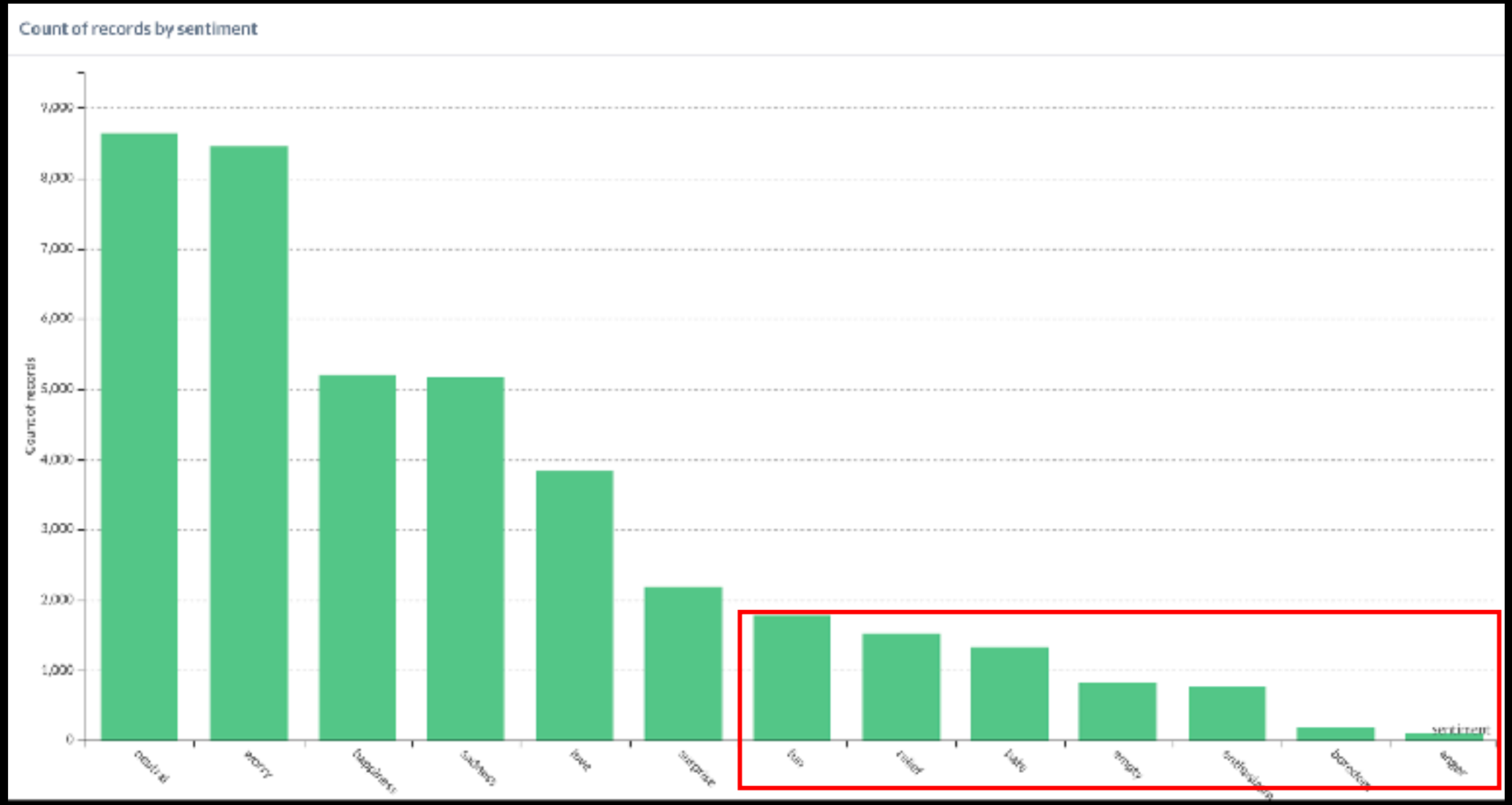
Dataset

- 40,000 tweets
- labelled with 14 emotion classes
- By CrowdFlower, hosted on data.world

Dataset


EMOTION &
FEELING WHEEL
Anger
Sadness
Surprise
Joy
Love
Fear
Relieved
Compassionate
Tenderness
Peaceful
Scared
Frightened
Stunned
Shocked
Confused
Despair
Powerless
Dismayed
Proud
Optimistic
Cheerful

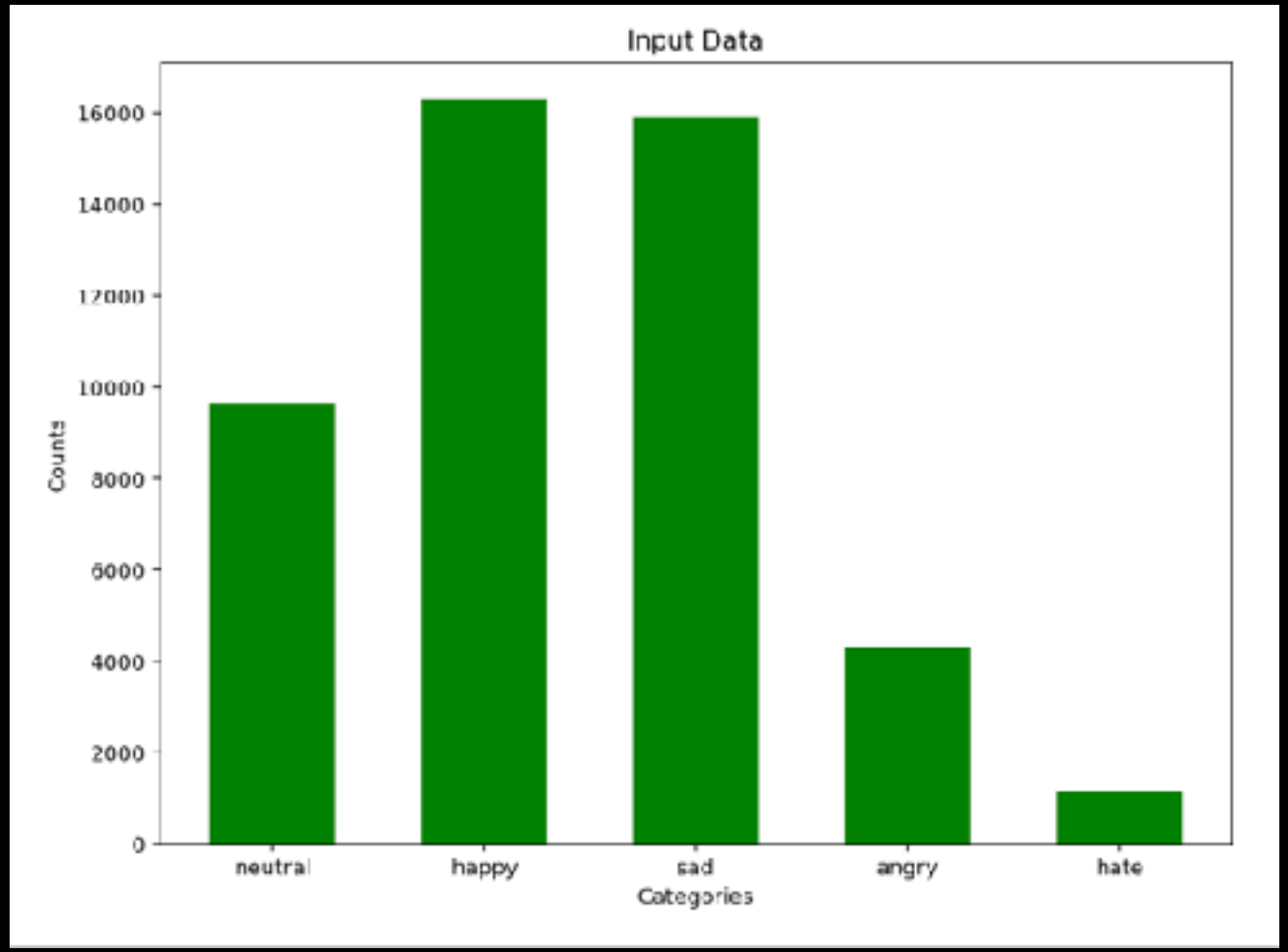
Dataset


Count of records by sentiment
Count of records
9,000
8,000
7,000
6,000
5,000
4,000
3,000
2,000
1,000
0
neutral
worry
happiness
sadness
love
surprise
fun
relief
hate
empty
enthusiasm
boredom
anger
sentiment

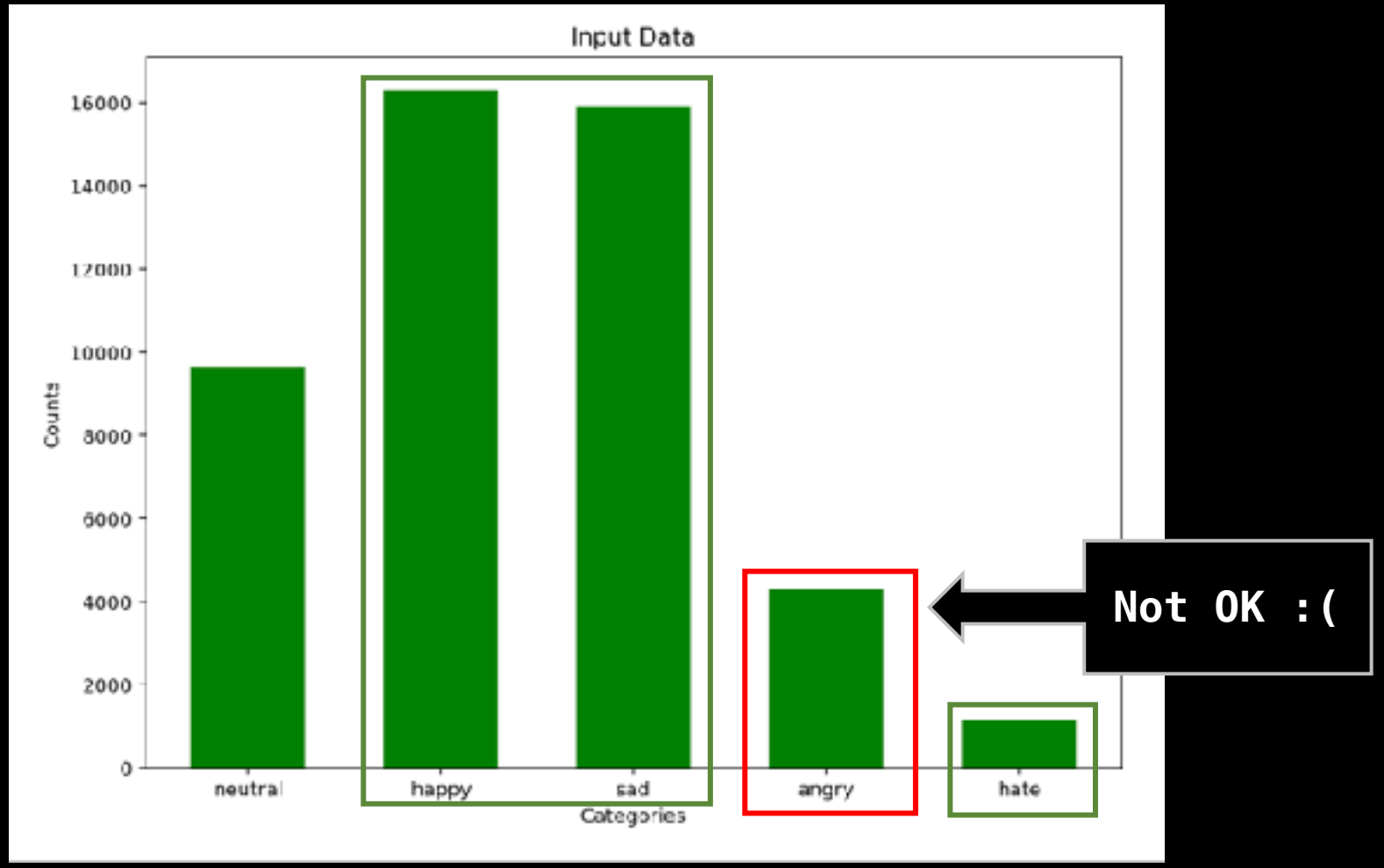
Data Engineering

- Merging of classes
  - Match previous papers (Bouazizi and Ohtsuki, 2017)
  - 5 classes: Neutral, Happy, Sad, Angry, Hate
- Acquire additional data
  - Crawl Twitter for tweets with ”tagged” emotions i.e. #happy #angry etc.
  - Not perfect :(
  - After some rejection/cleaning: 47,288 tweets

Data Engineering



Data Engineering



Applying Deep Learning

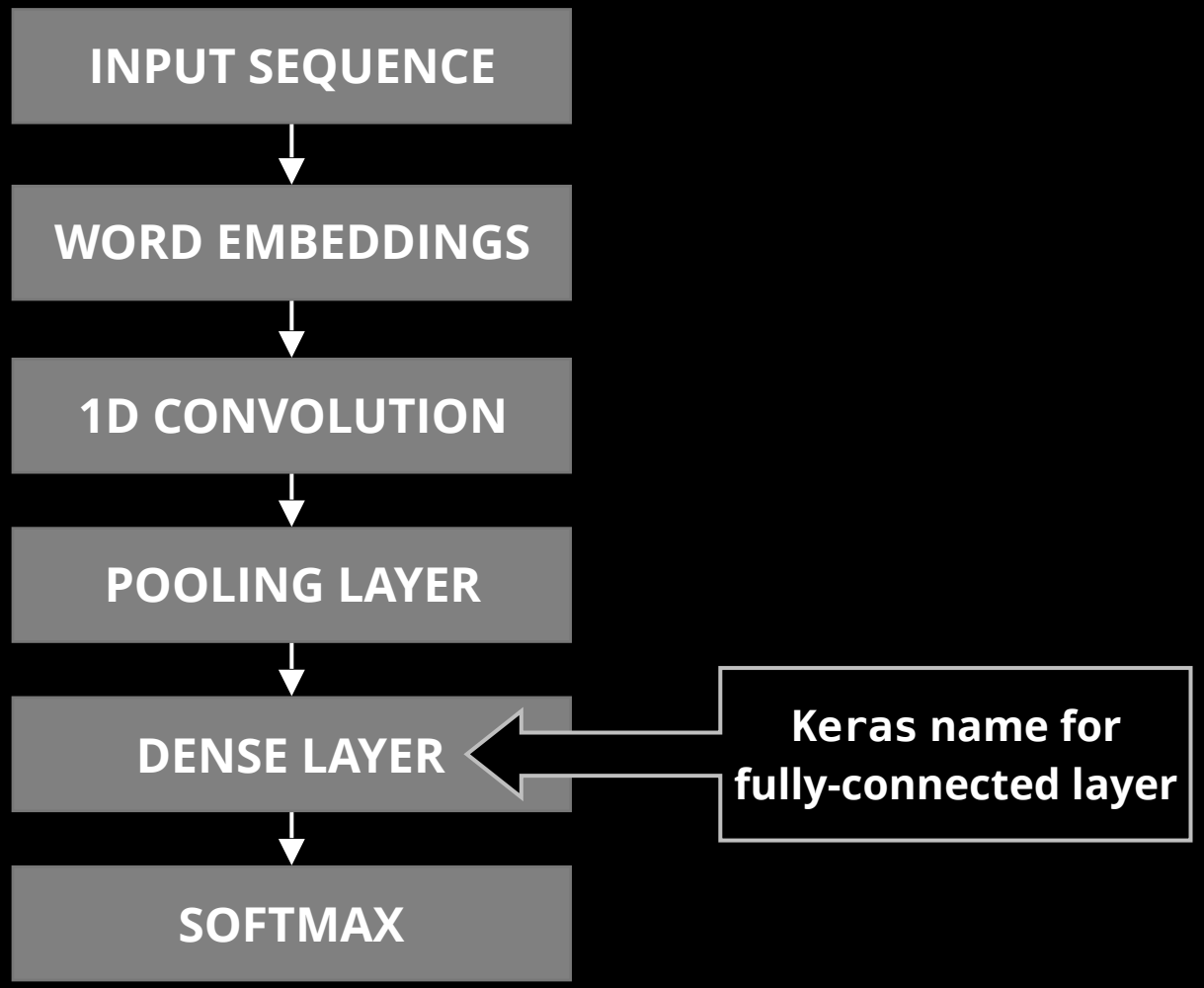
Artificial Neural Networks

Model Design

- Extract as many **features**/representations as possible
  - Lack of data (in length of text and number of examples per class)
- Be as **generalisable** as possible
  - Work on Singlish and different type of source medium
  - Target application is comments on local news websites and blogs
    - As opposed to tweets in training data

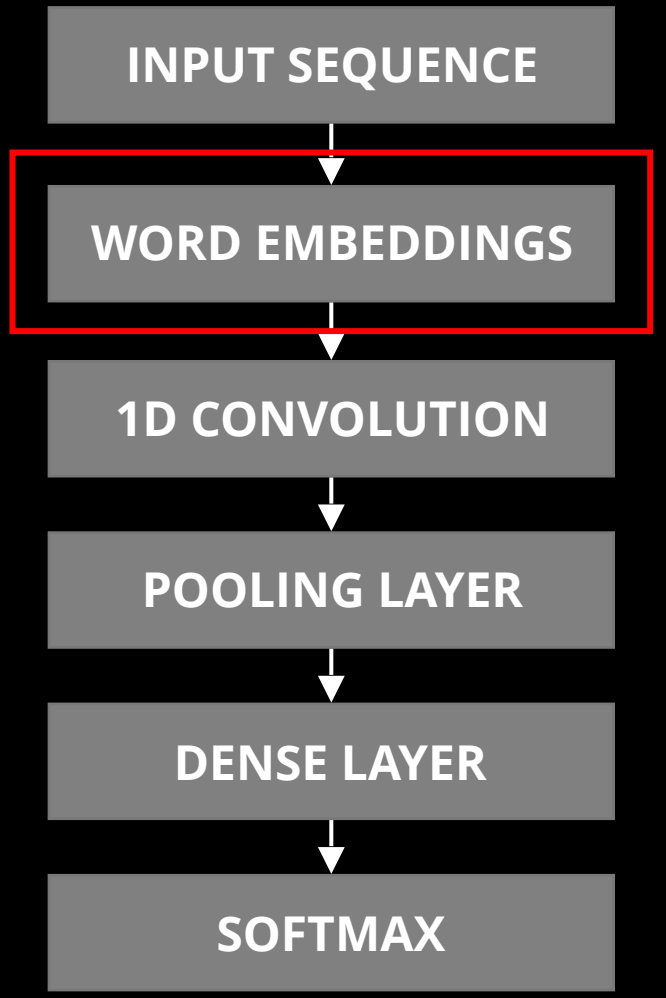
Model Design

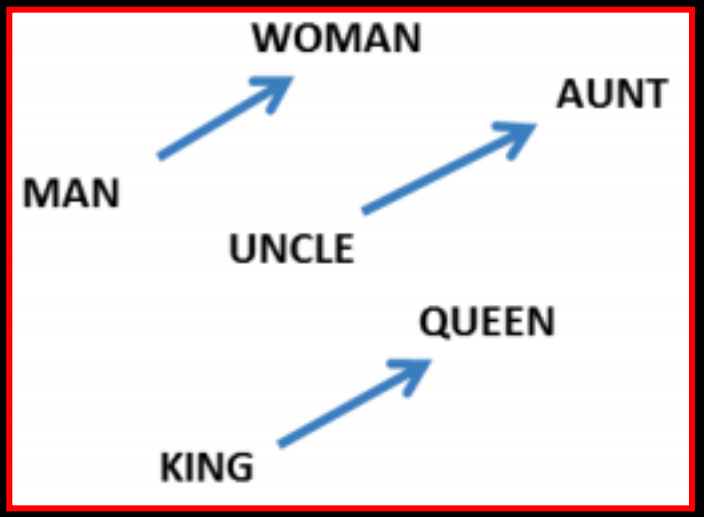
"Vanilla" CNN



Model Design

"Vanilla" CNN





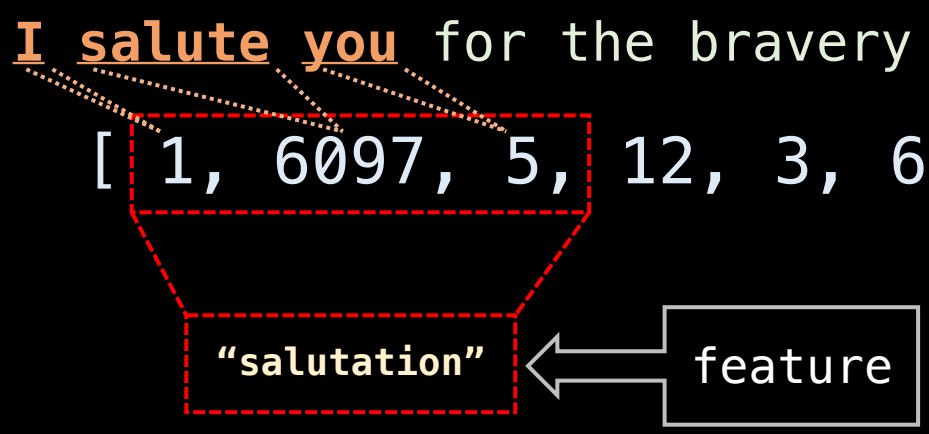
High dimensional vector space that captures relationships between terms

CNN applied to words

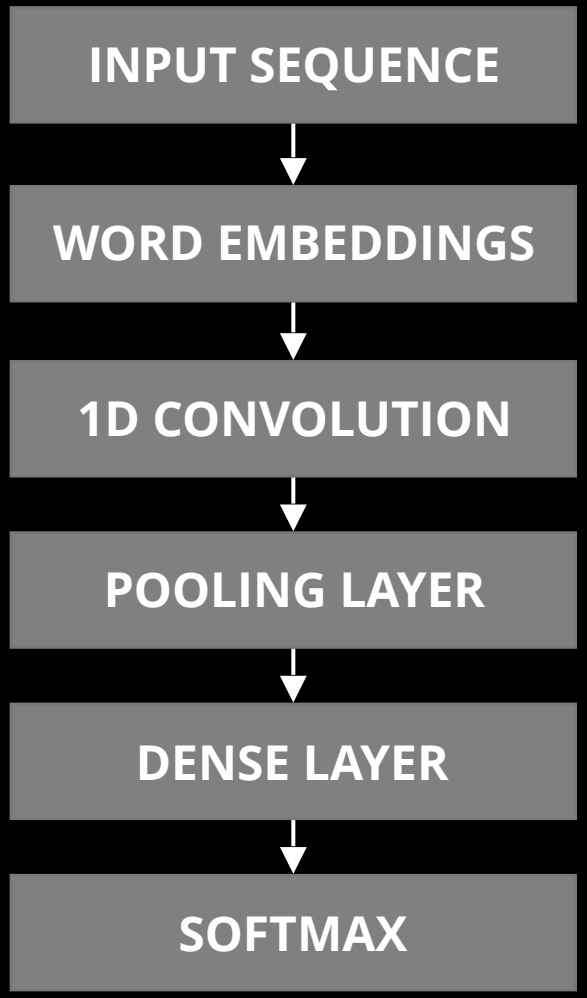
- Extract **features** using 1D convolution

CNN applied to words

- Extract **features** using 1D convolution

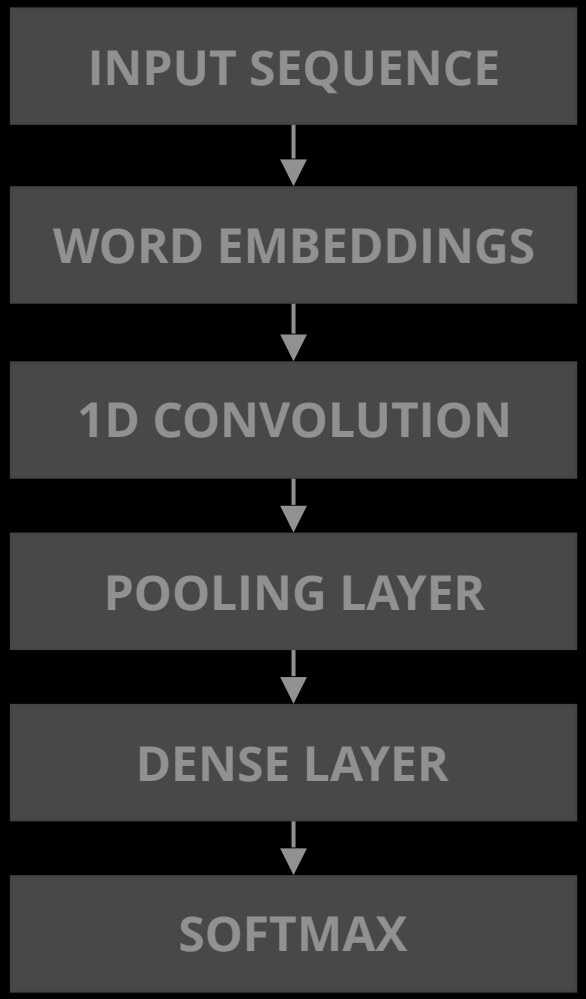


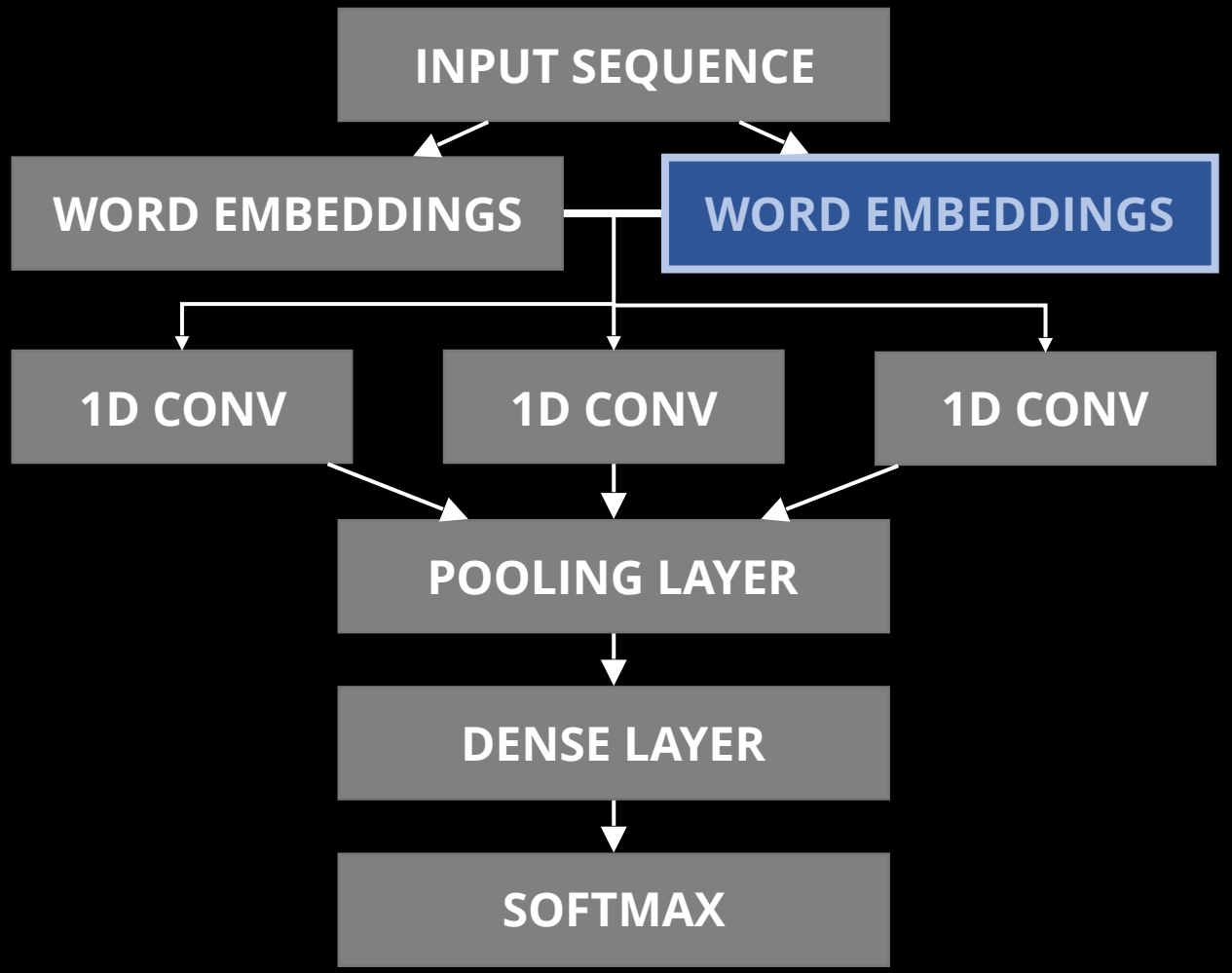
Model Design



Model Design

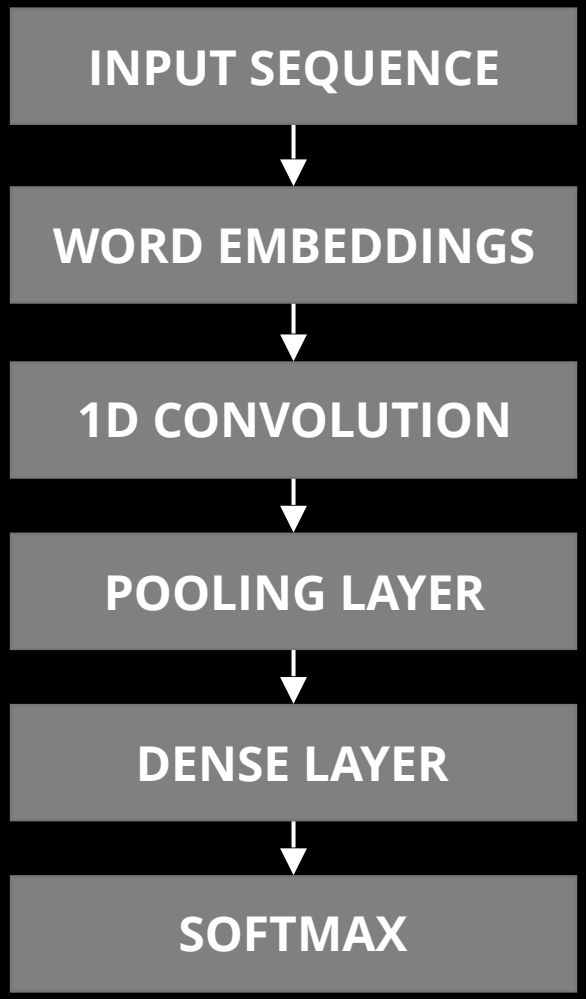
Yoon Kim, 2014

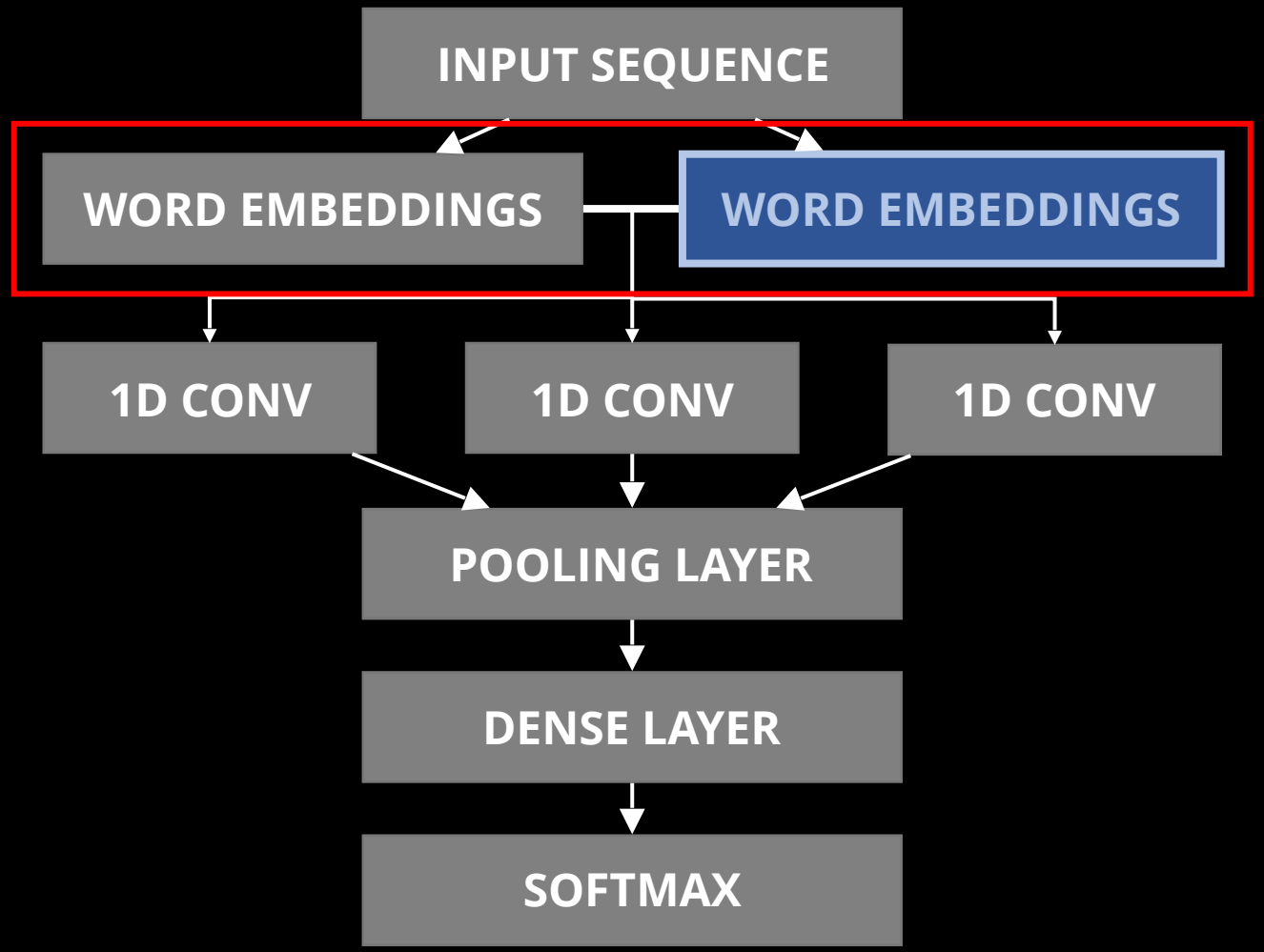




Model Design

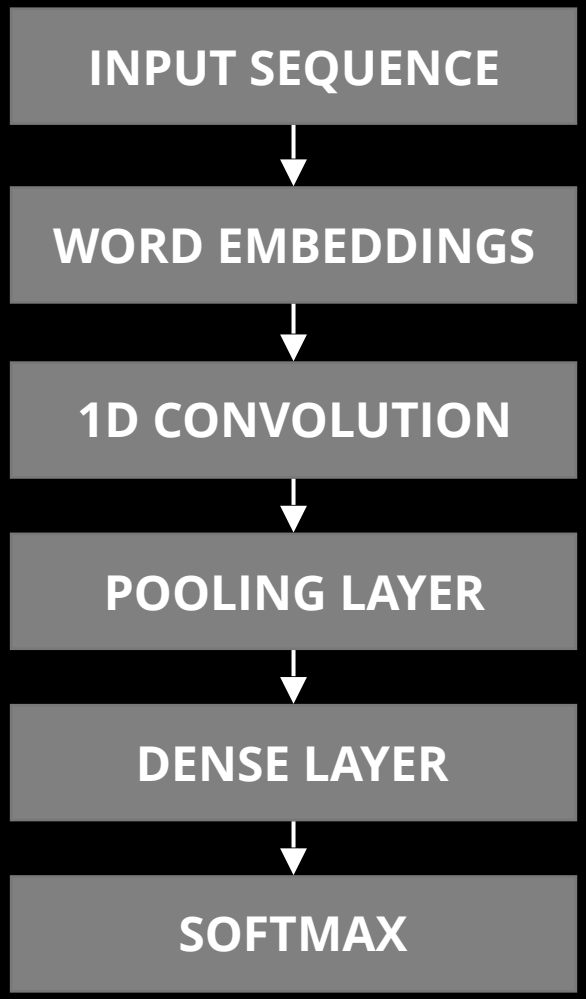
Yoon Kim, 2014

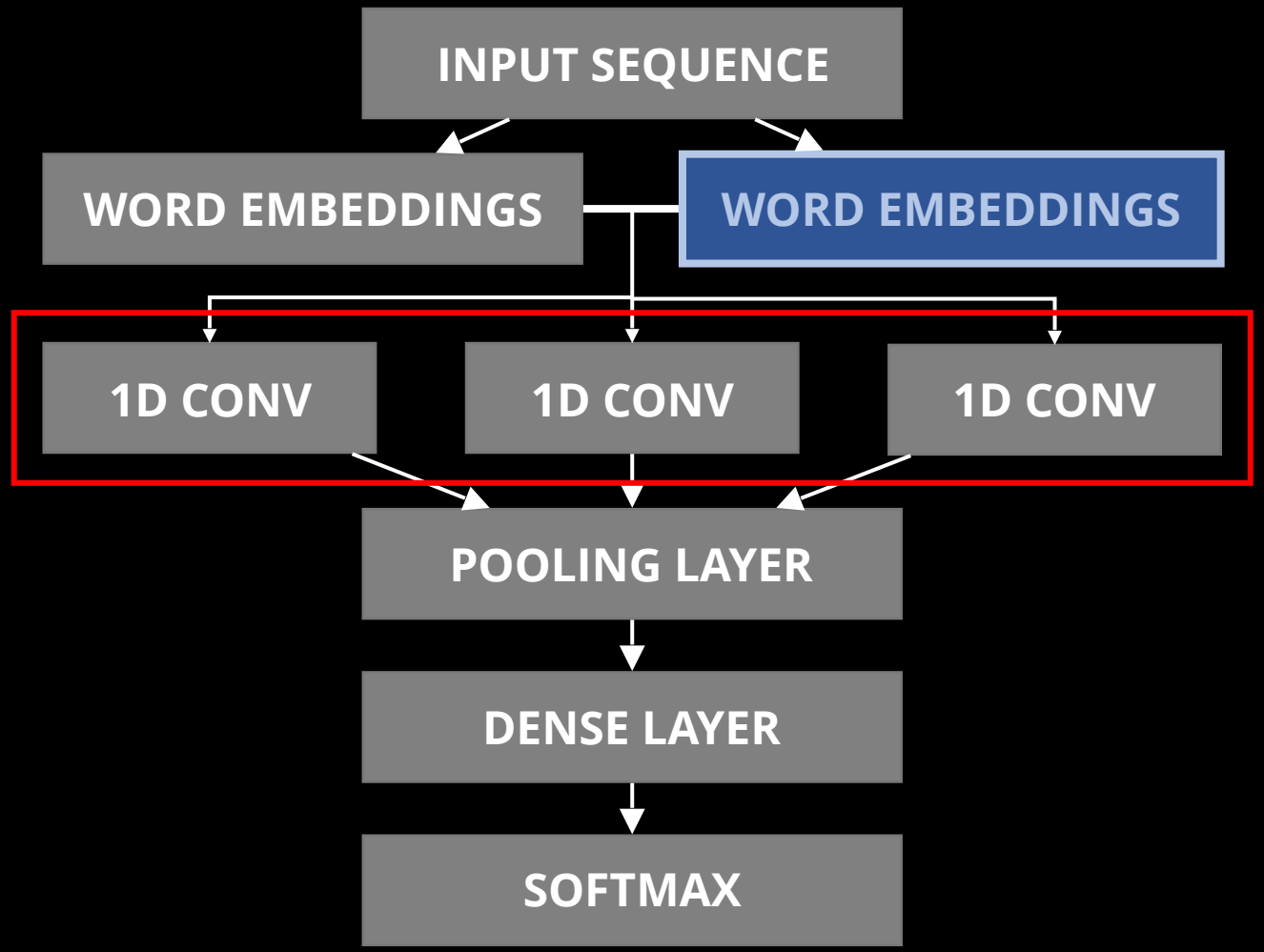




Model Design

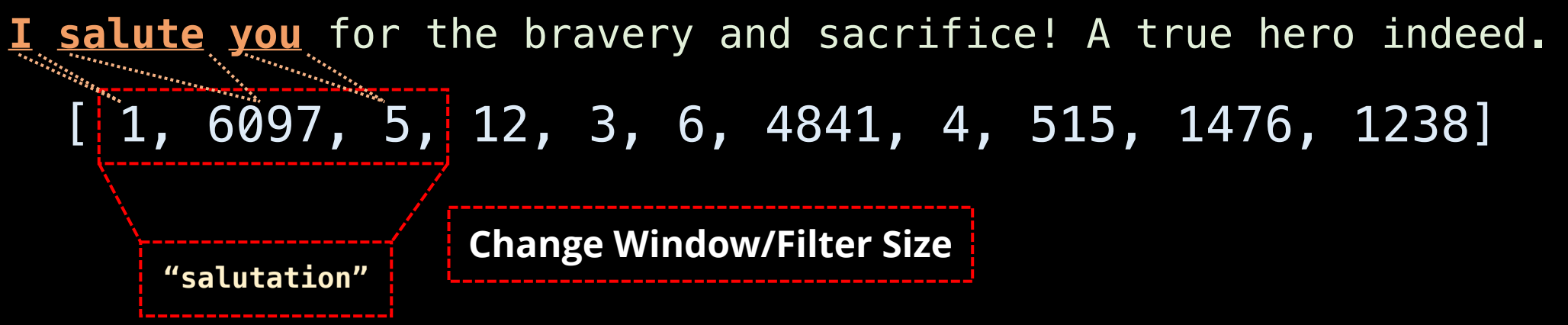
Yoon Kim, 2014





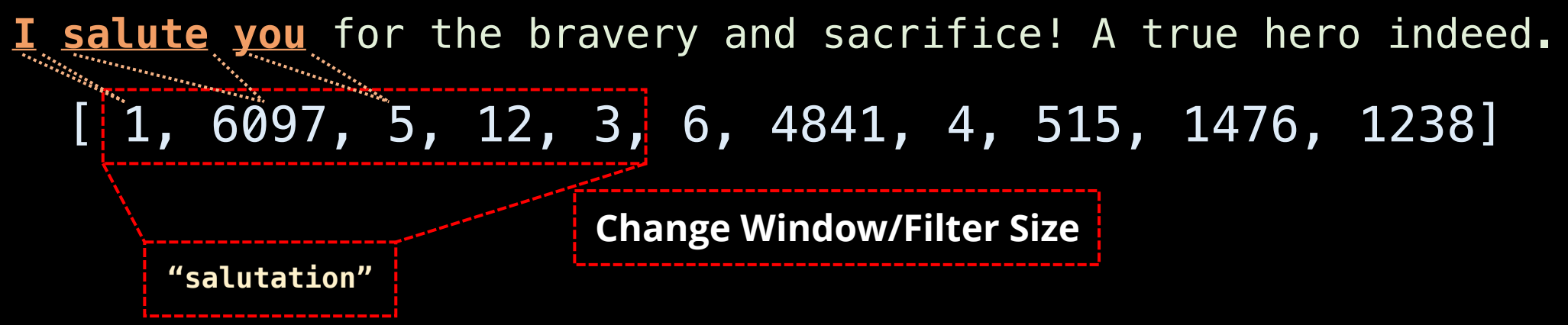
CNN applied to words

- Extract **features** using 1D convolution



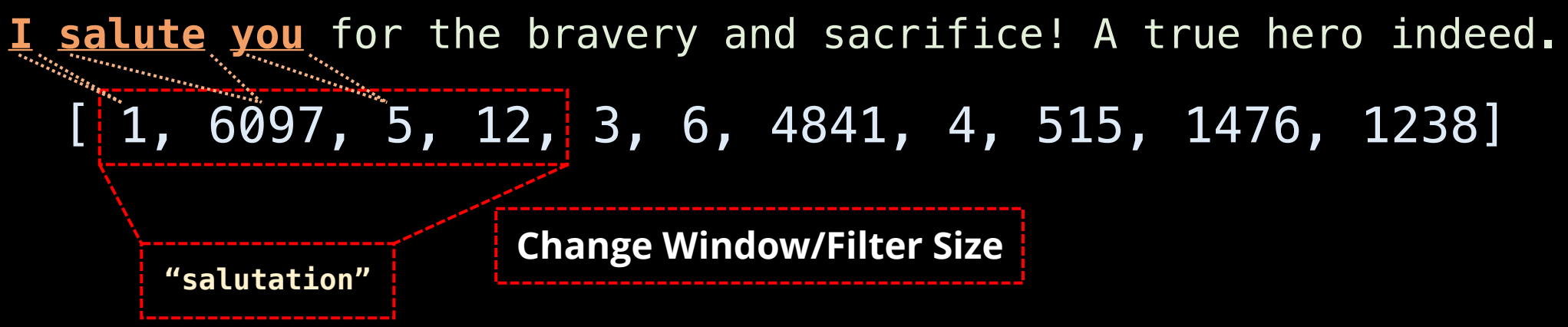
CNN applied to words

- Extract **features** using 1D convolution



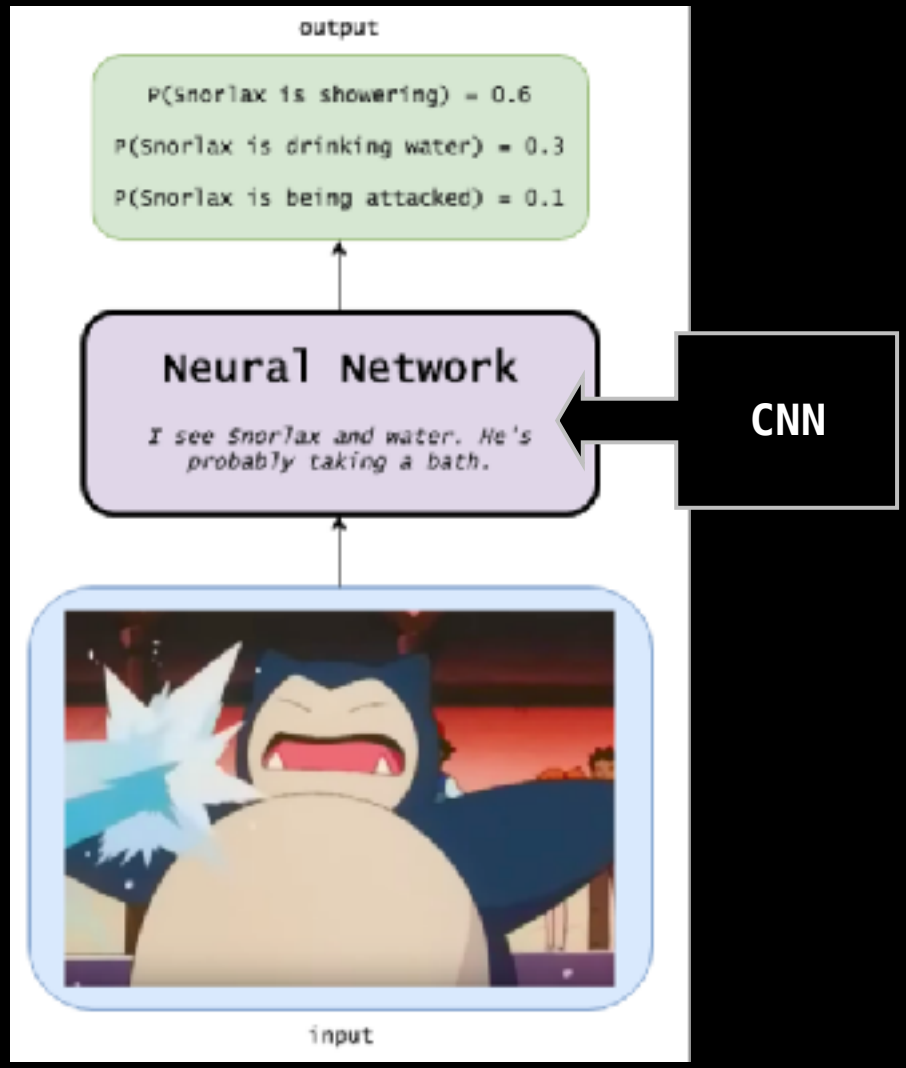
CNN applied to words

- Extract **features** using 1D convolution



Downside of CNN

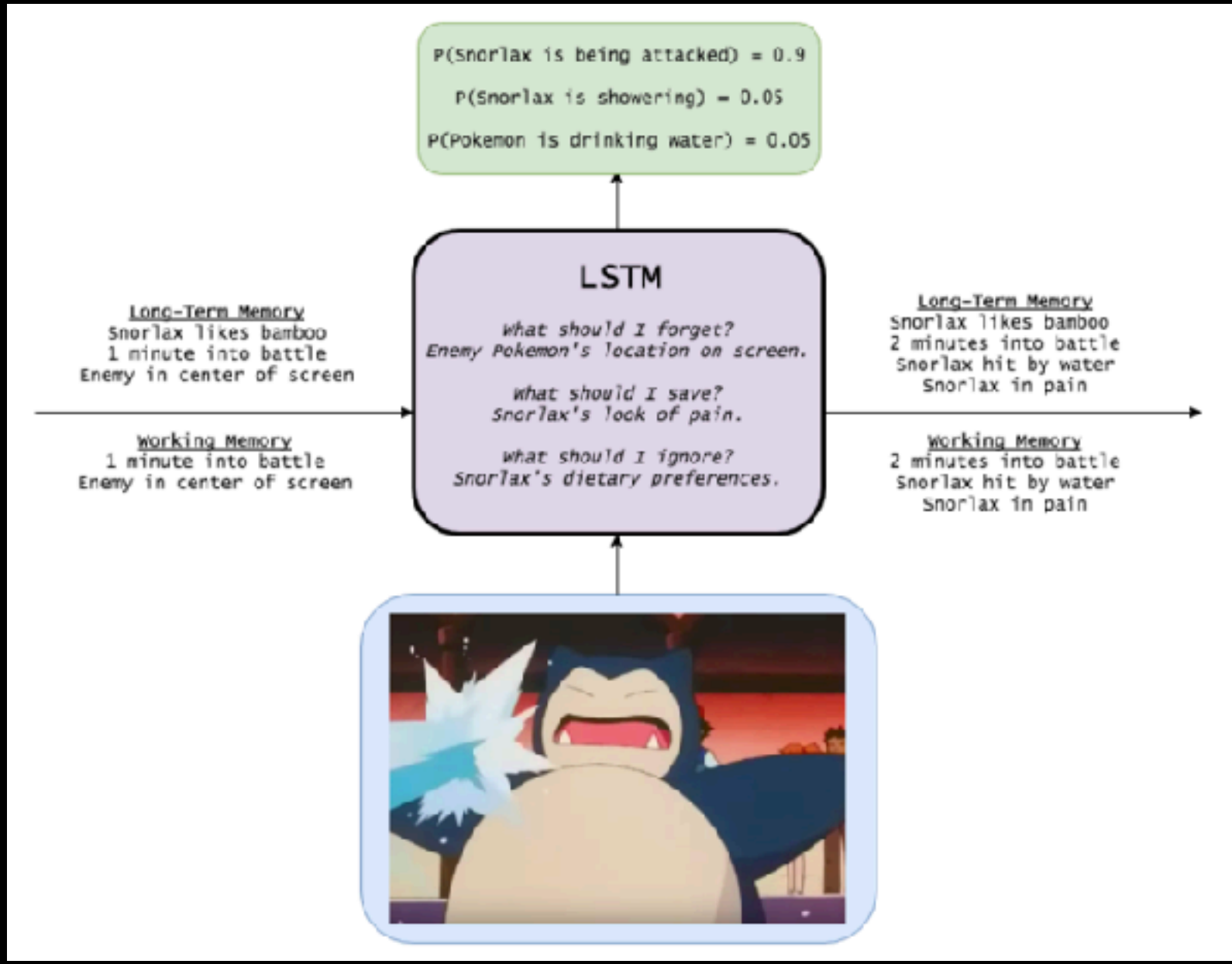
- Sequence matters
- Context is important



LSTM

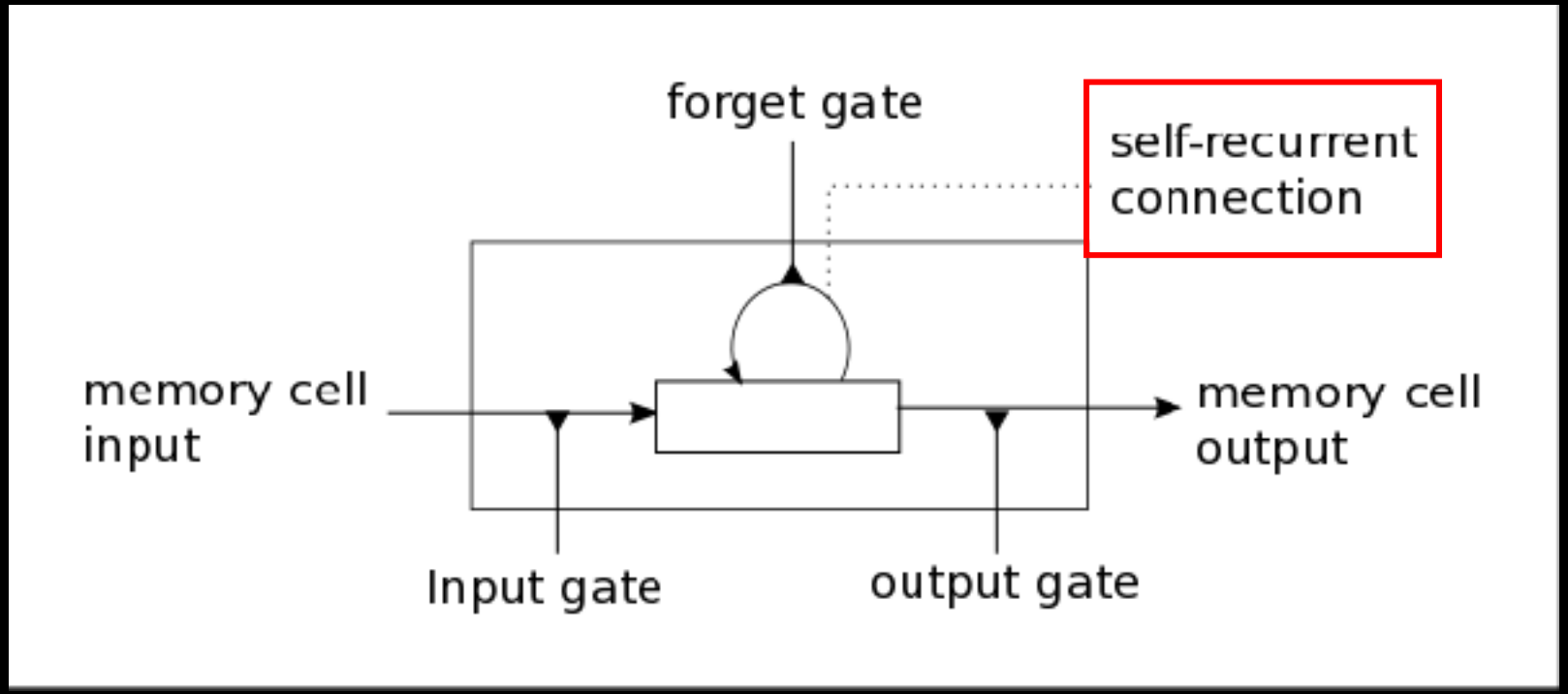
- Type of recurrent neural network (RNN)
  - To learn **sequences** via series of hidden states
  - Memory cells can keep information intact
    - unless inputs makes them forget/overwrite
  - Cell can decide to output this information or just store it

LSTM


P(Snorlax is being attacked) = 0.9
P(Snorlax is showering) = 0.05
P(Pokemon is drinking water) = 0.05
LSTM
What should I forget?
Enemy Pokemon's location on screen.
What should I save?
Snorlax's lack of pain.
What should I ignore?
Snorlax's dietary preferences.
Long-Term Memory
Snorlax likes bamboo
1 minute into battle
Enemy in center of screen
Working Memory
1 minute into battle
Enemy in center of screen
Long-Term Memory
Snorlax likes bamboo
2 minutes into battle
Snorlax hit by water
Snorlax in pain
Working Memory
2 minutes into battle
Snorlax hit by water
Snorlax in pain

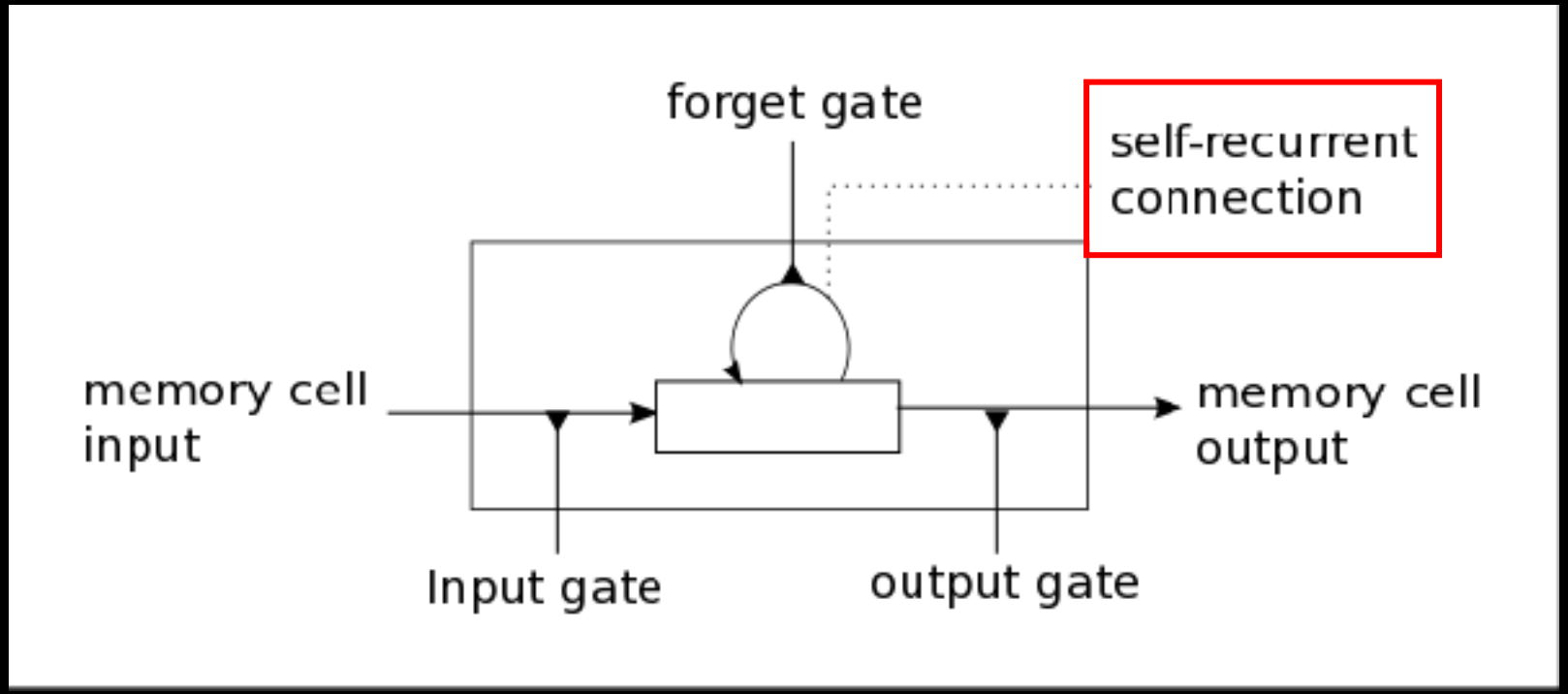
LSTM

```
[ 1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]
```



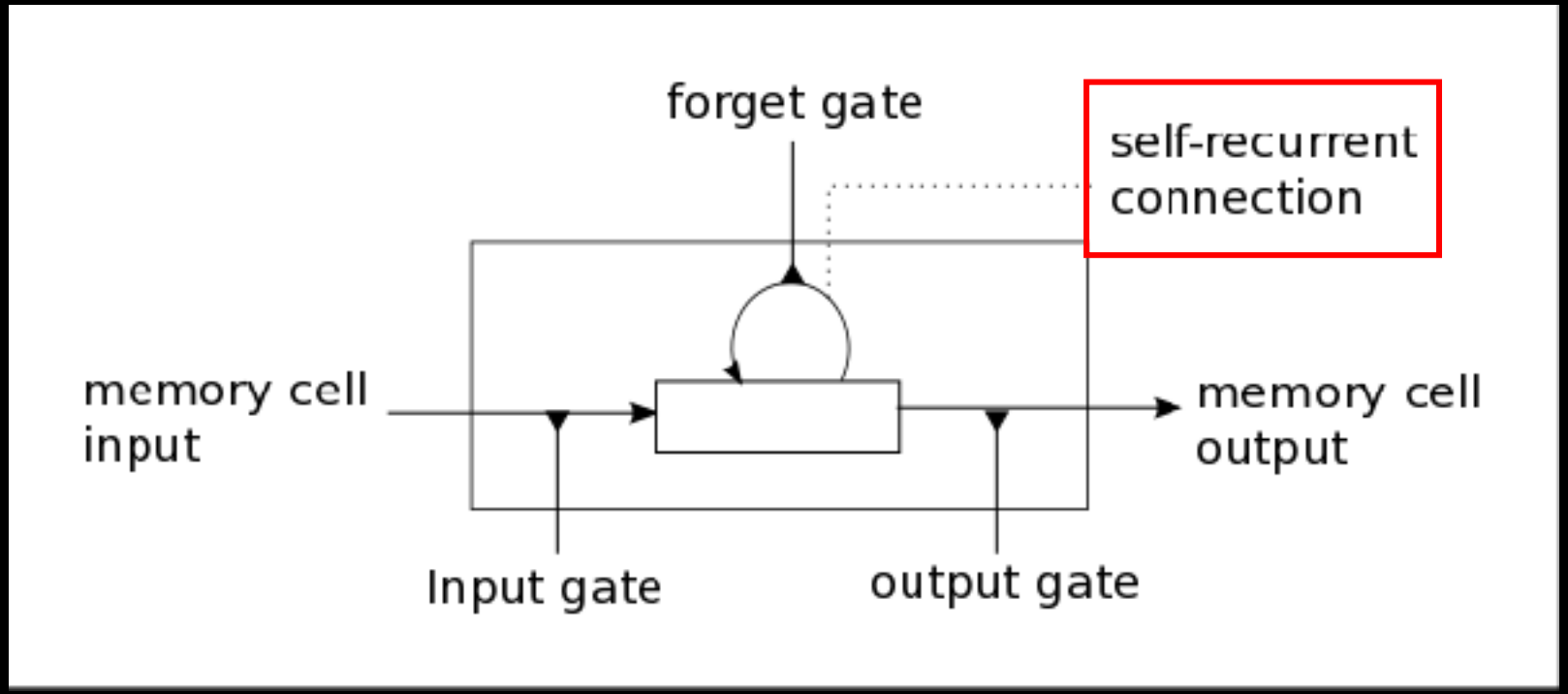
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



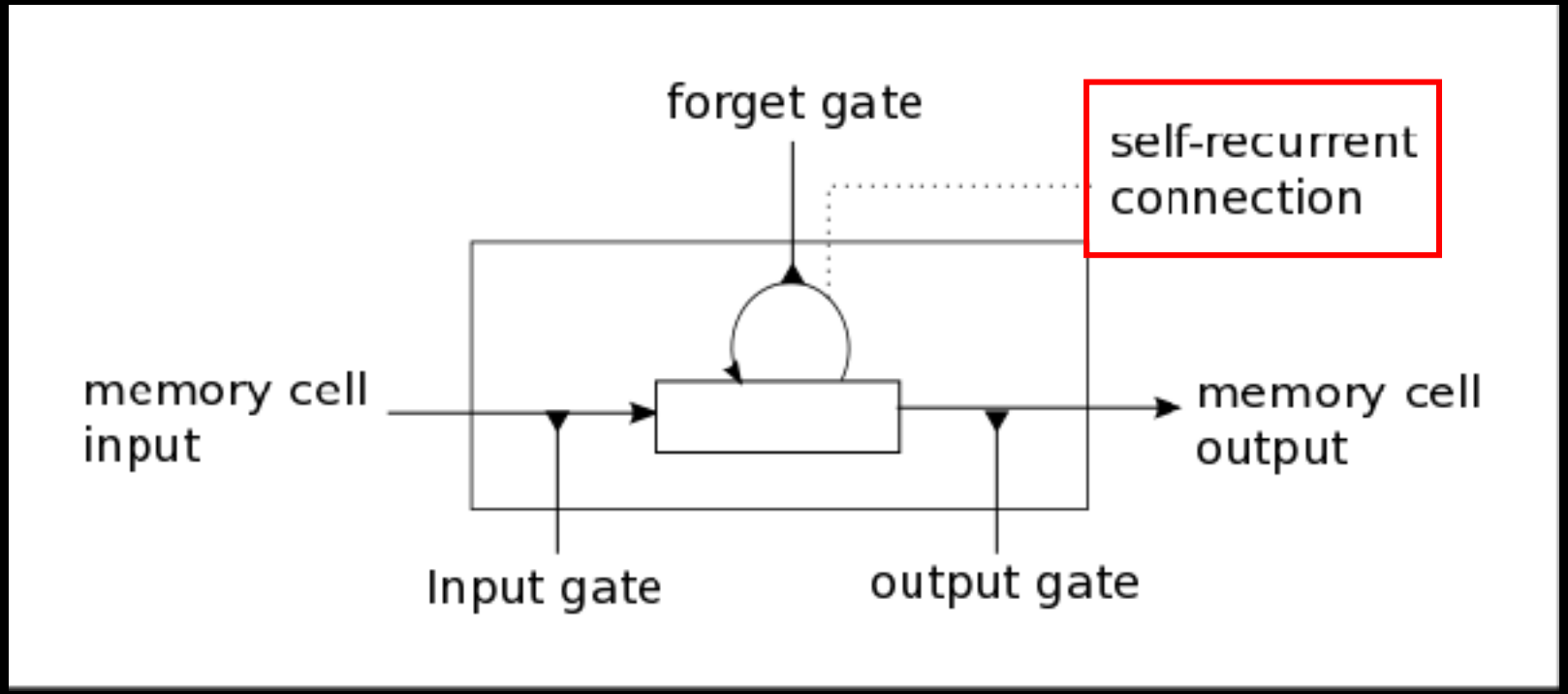
LSTM

```
[ 1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]
```



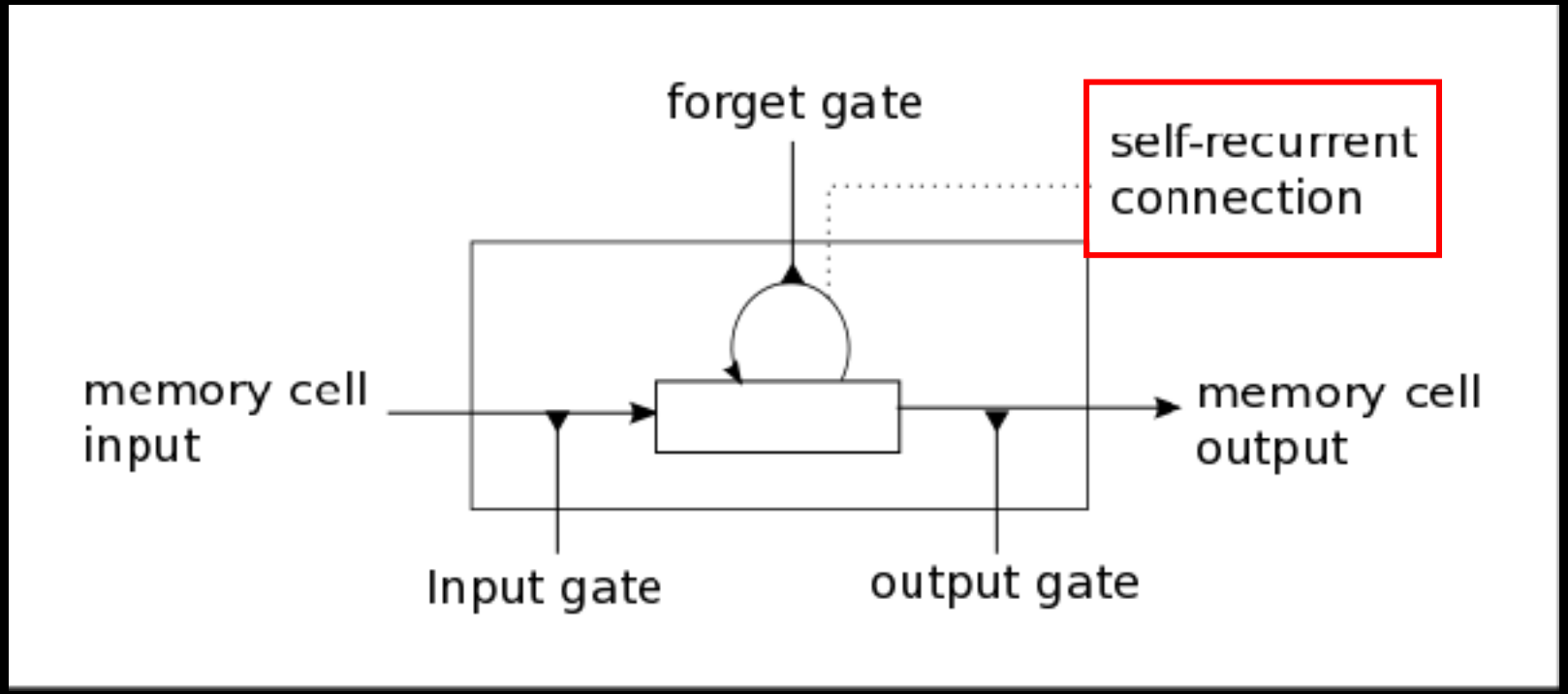
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



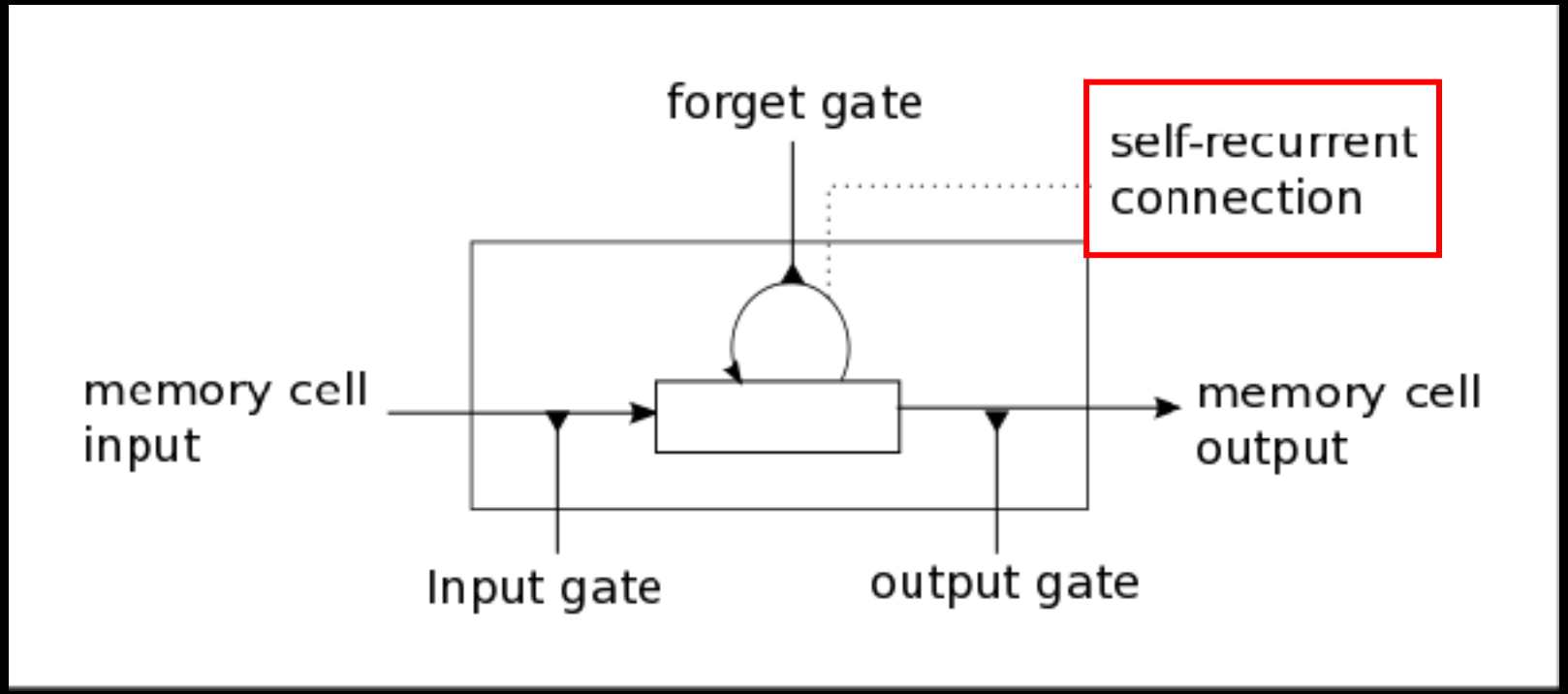
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



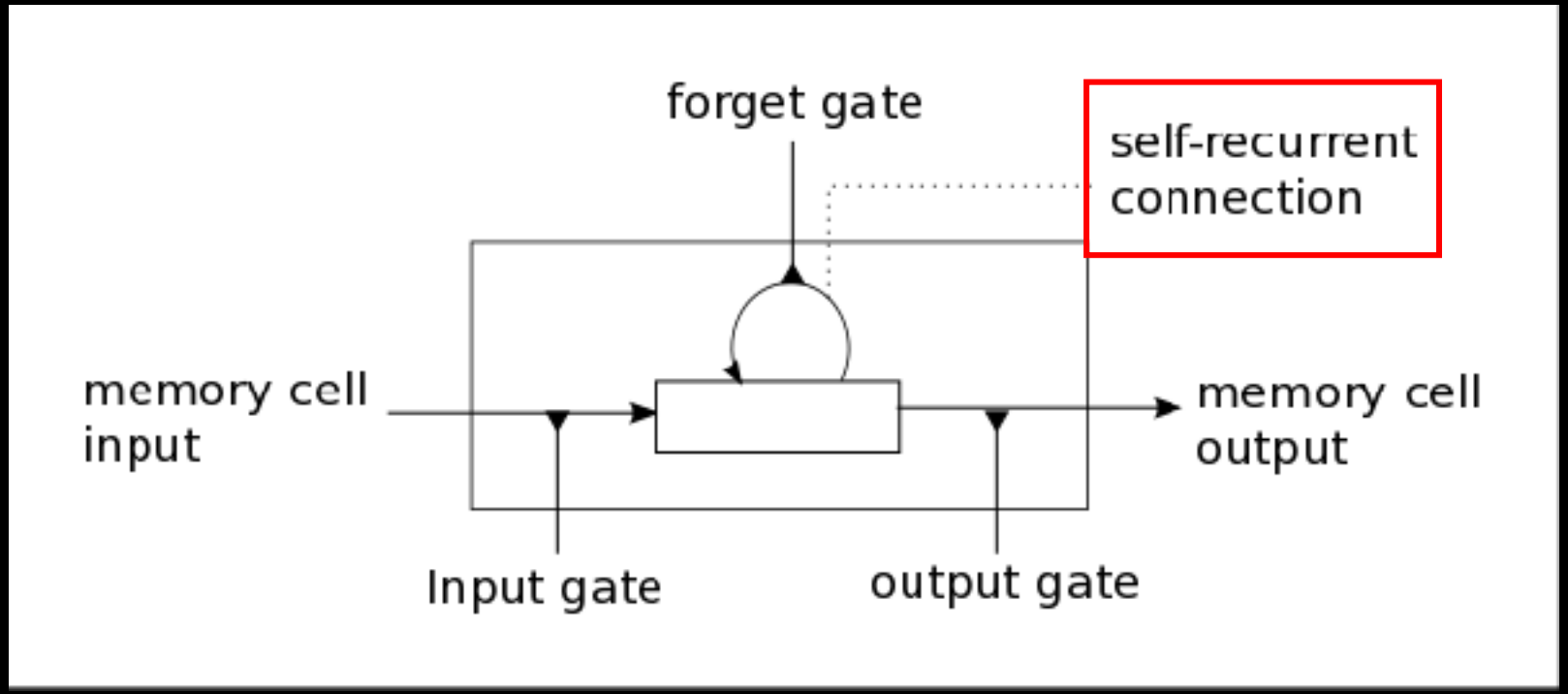
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



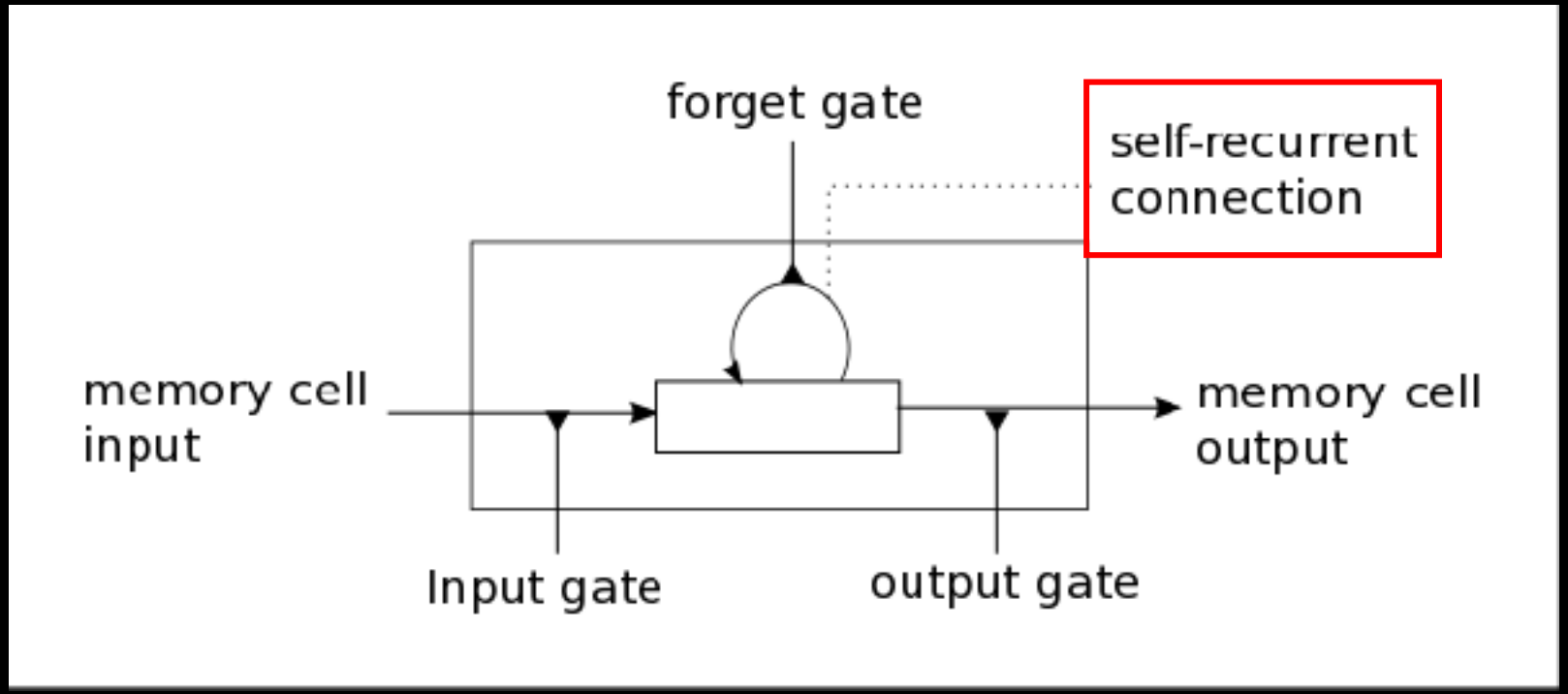
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



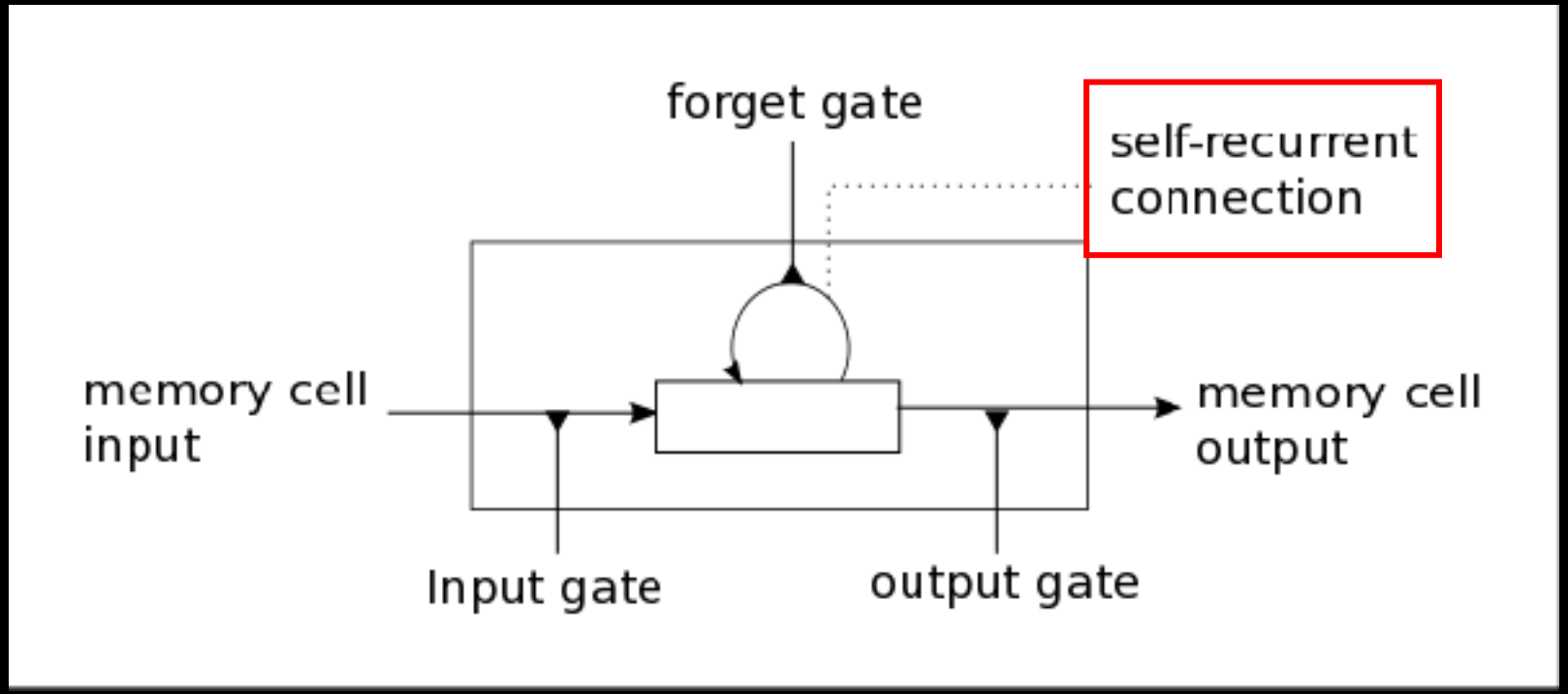
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



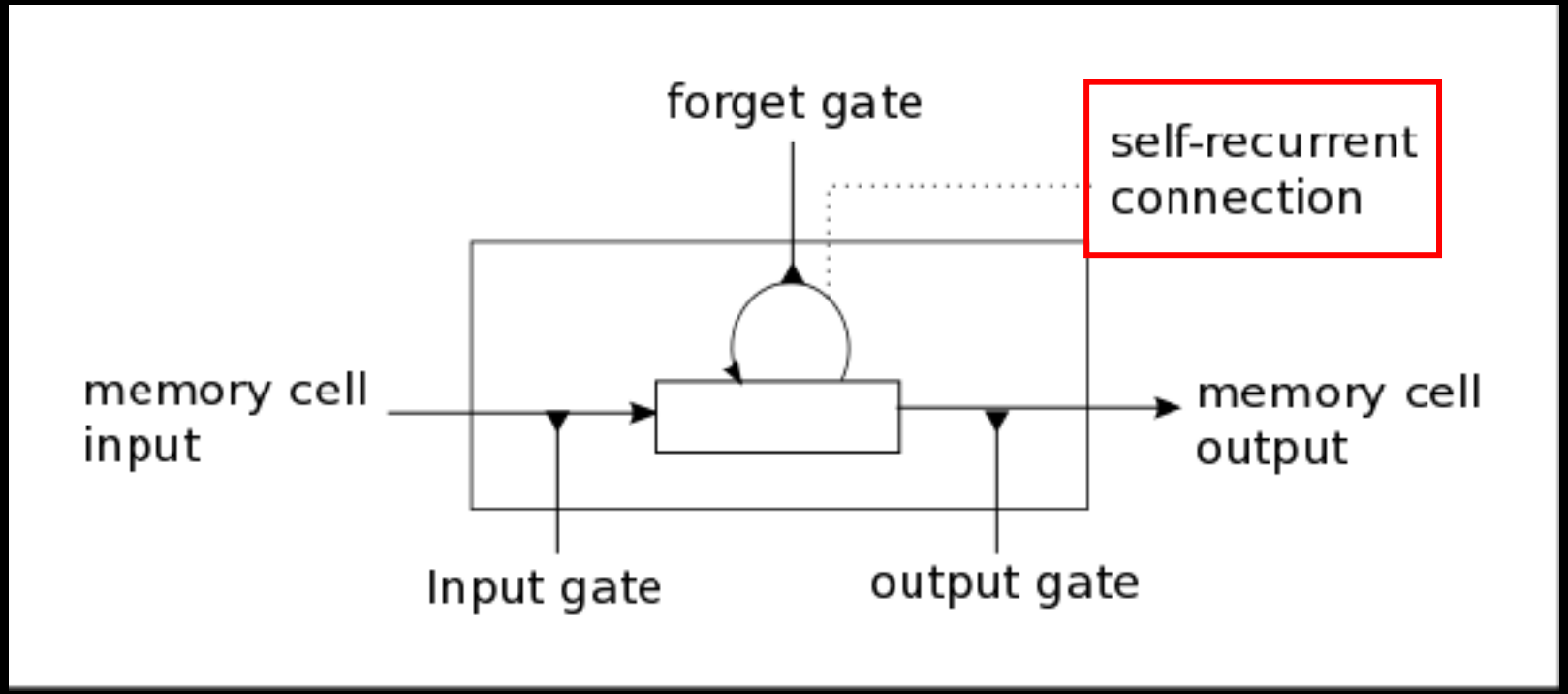
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



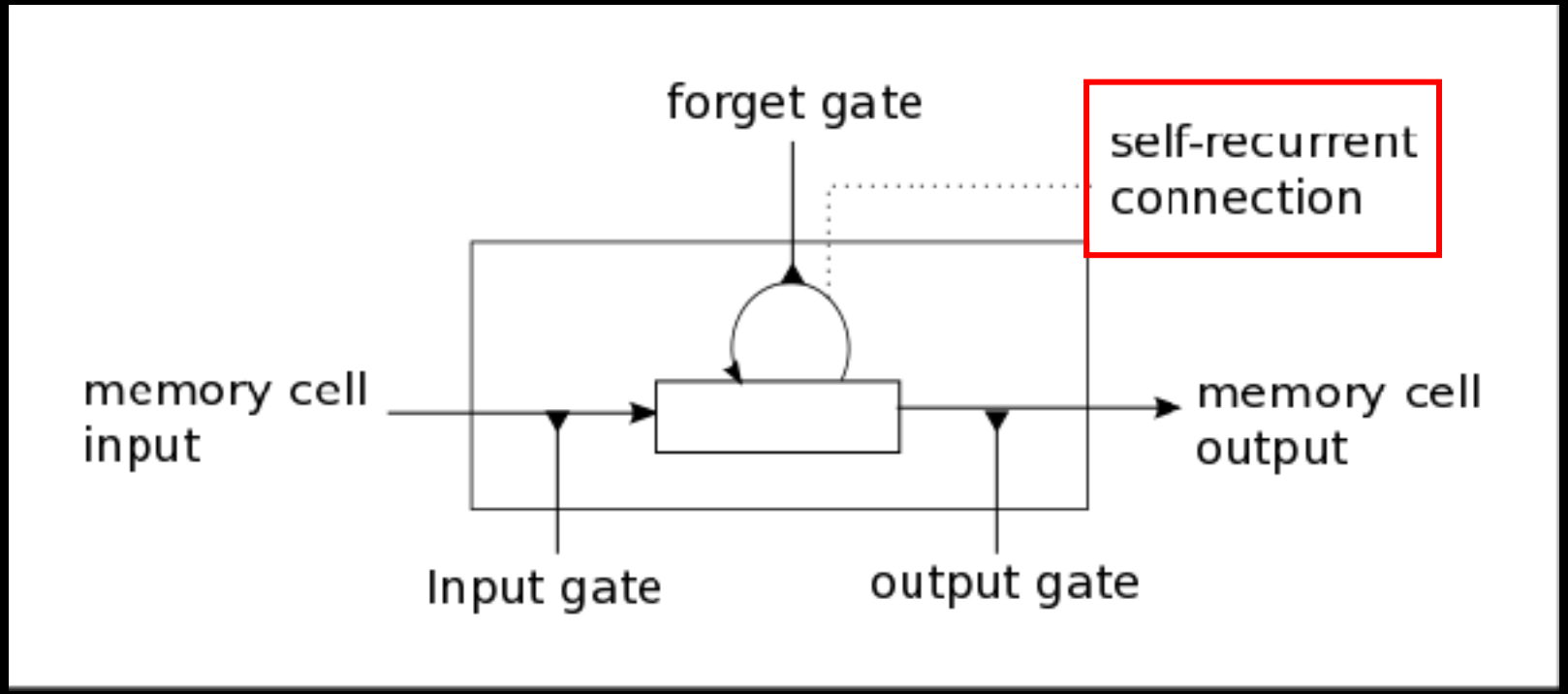
LSTM

[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



LSTM

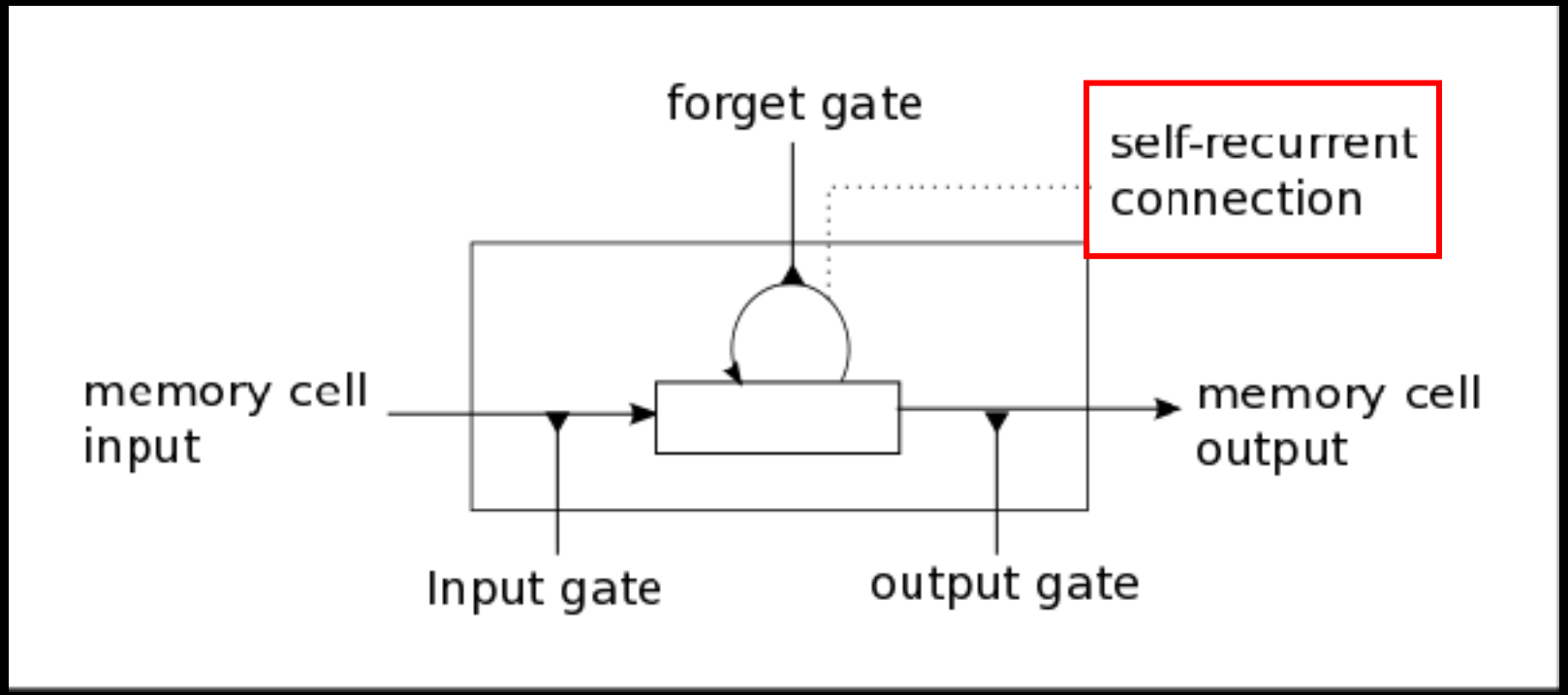
[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]



LSTM

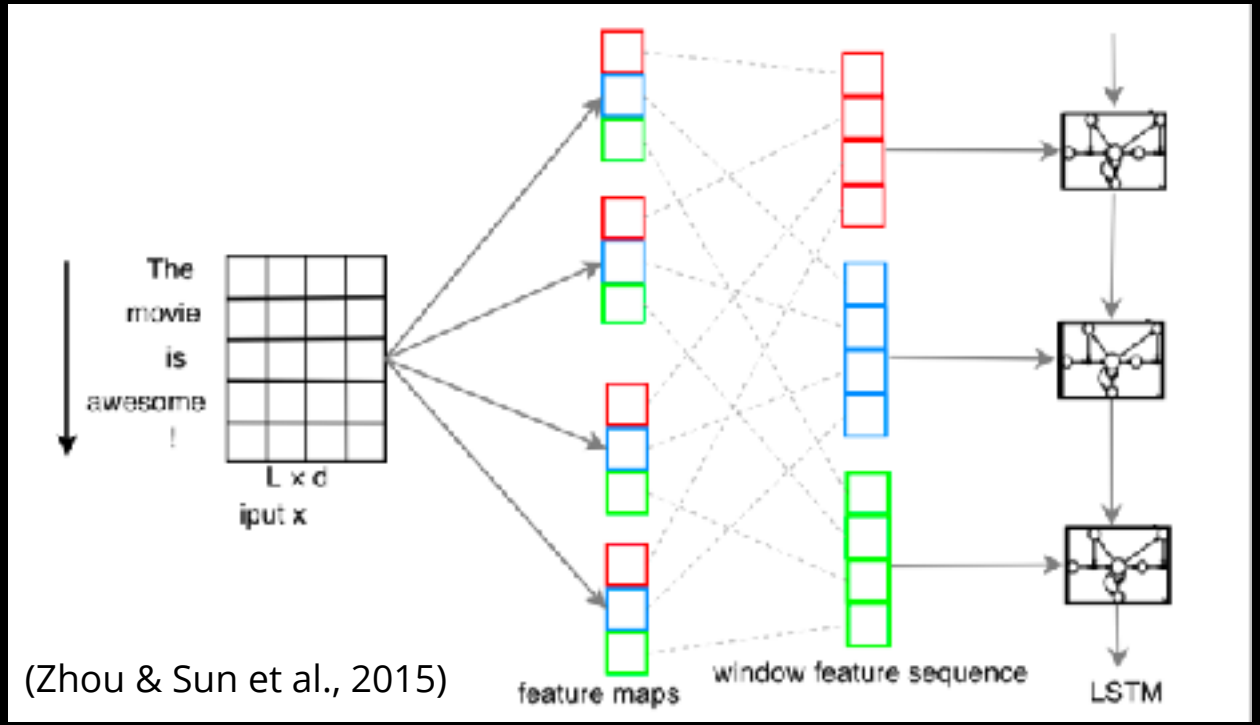
[1, 6097, 5, 12, 3, 6, 4841, 4, 515, 1476, 1238]

output



Model Design

- Why should we combine CNN and LSTM?
  - To learn **sequences of features** and **features of sequences**.

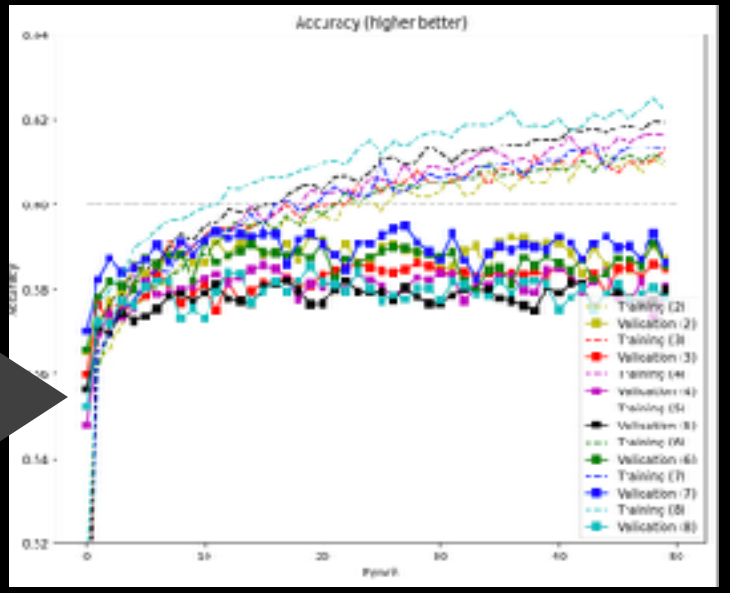


(Zhou & Sun et al., 2015)

Model Design

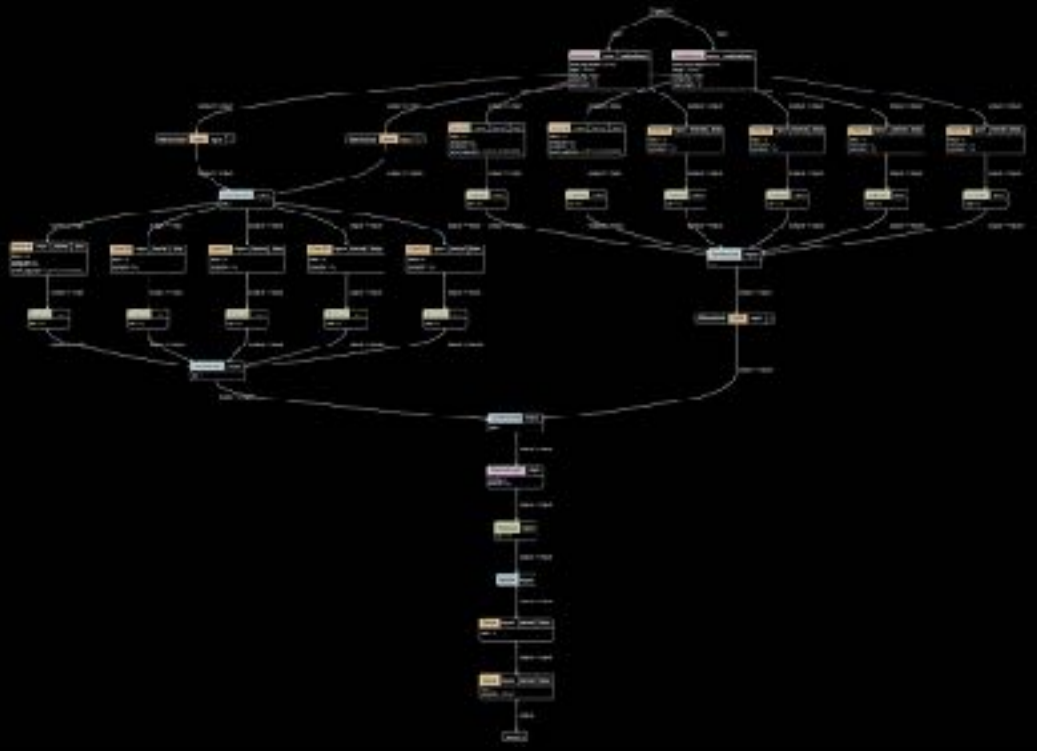
- Why should we combine CNN and LSTM?
  - To learn **sequences of features** and **features of sequences**.
- “Linear combination”: overall validation accuracy < 60%
  - How can we maximise feature extraction and improve classification accuracy to > 60%?
  - Hyper-parameter tuning not enough

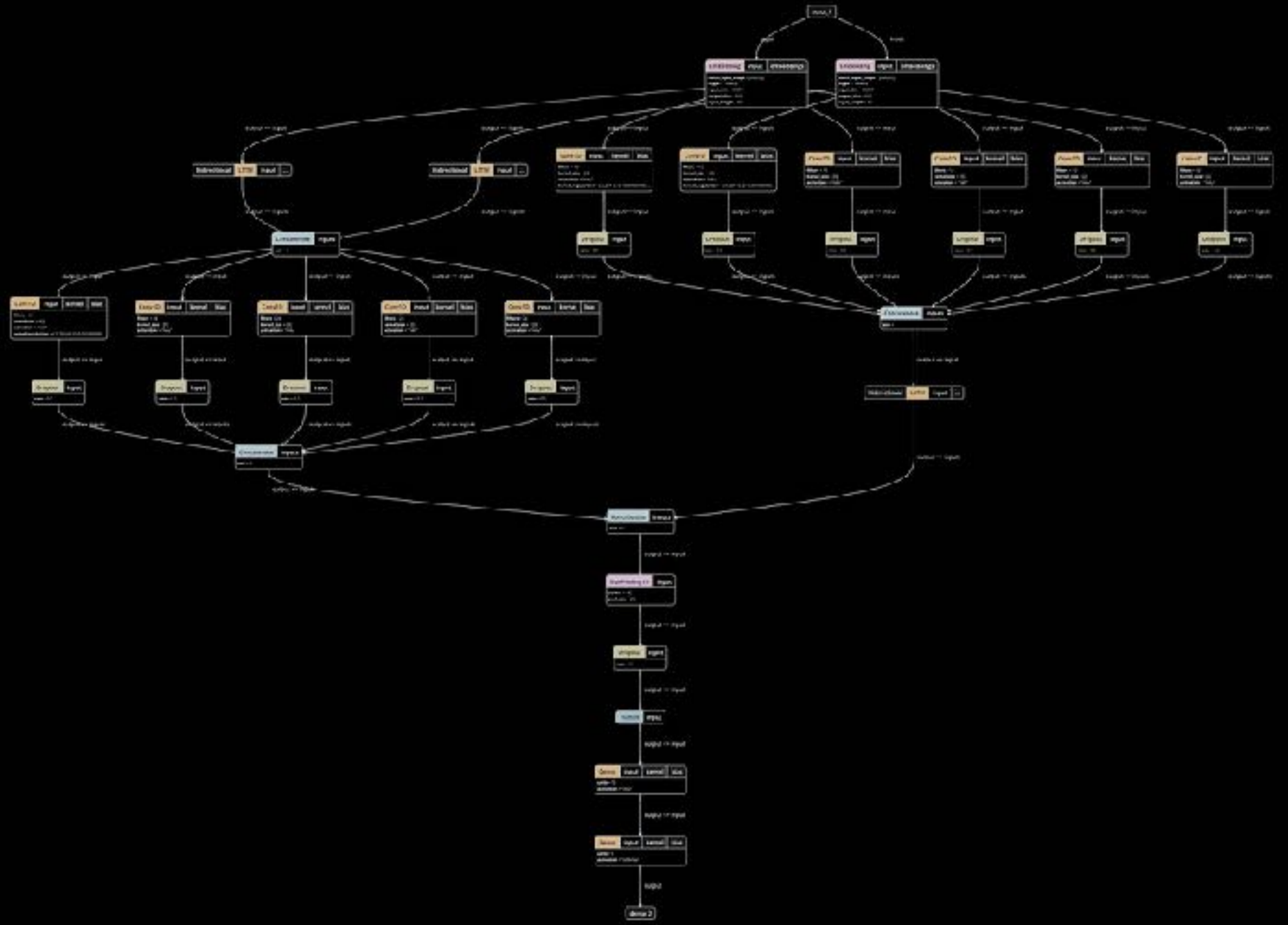
tried



BalanceNet

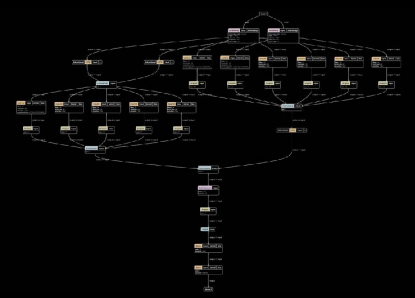
- Make use of an inter-connected architecture of “opposite” architectures
- It does look like a balance



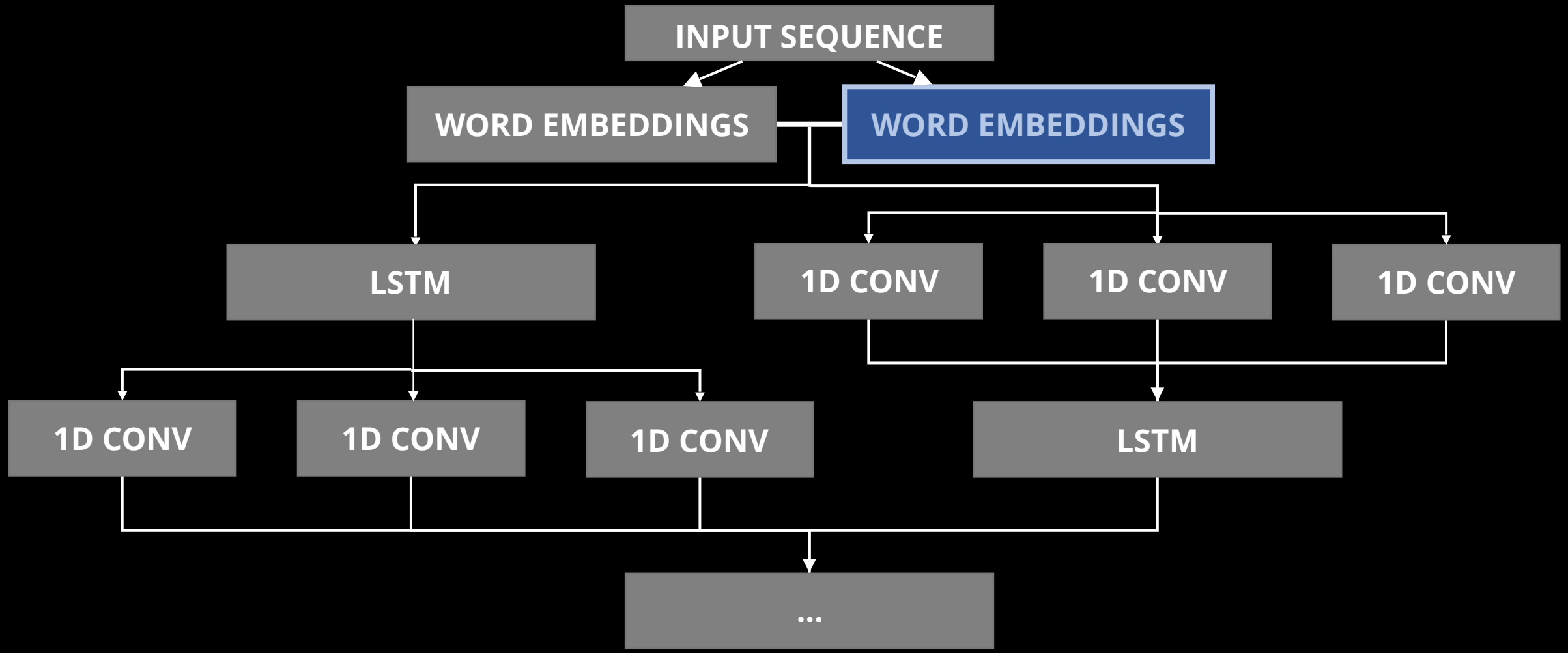


BalanceNet

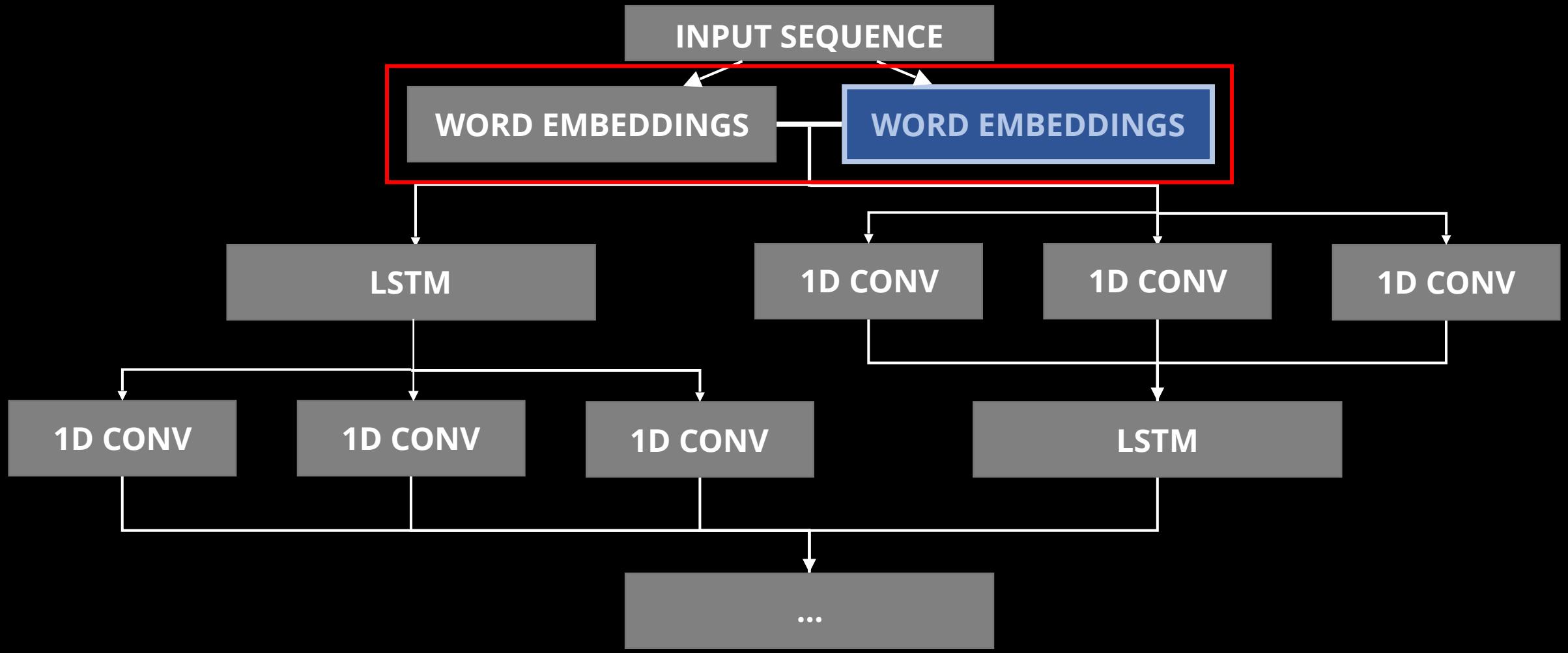
- Make use of an inter-connected architecture of “opposite” architectures
- Captures all previous ideas on how to design the model
- Create multiple “channels” through the model to allow the model to select which channels work better for certain classes



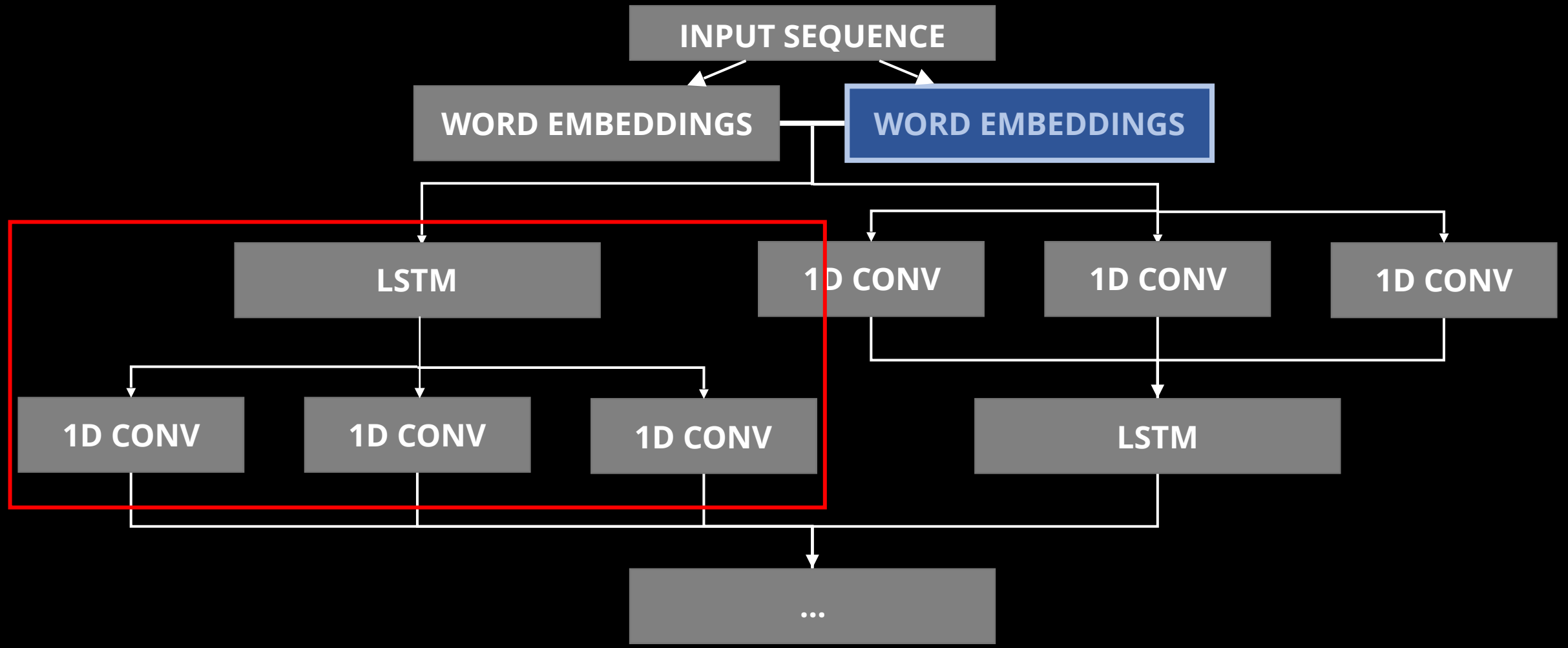
BalanceNet


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

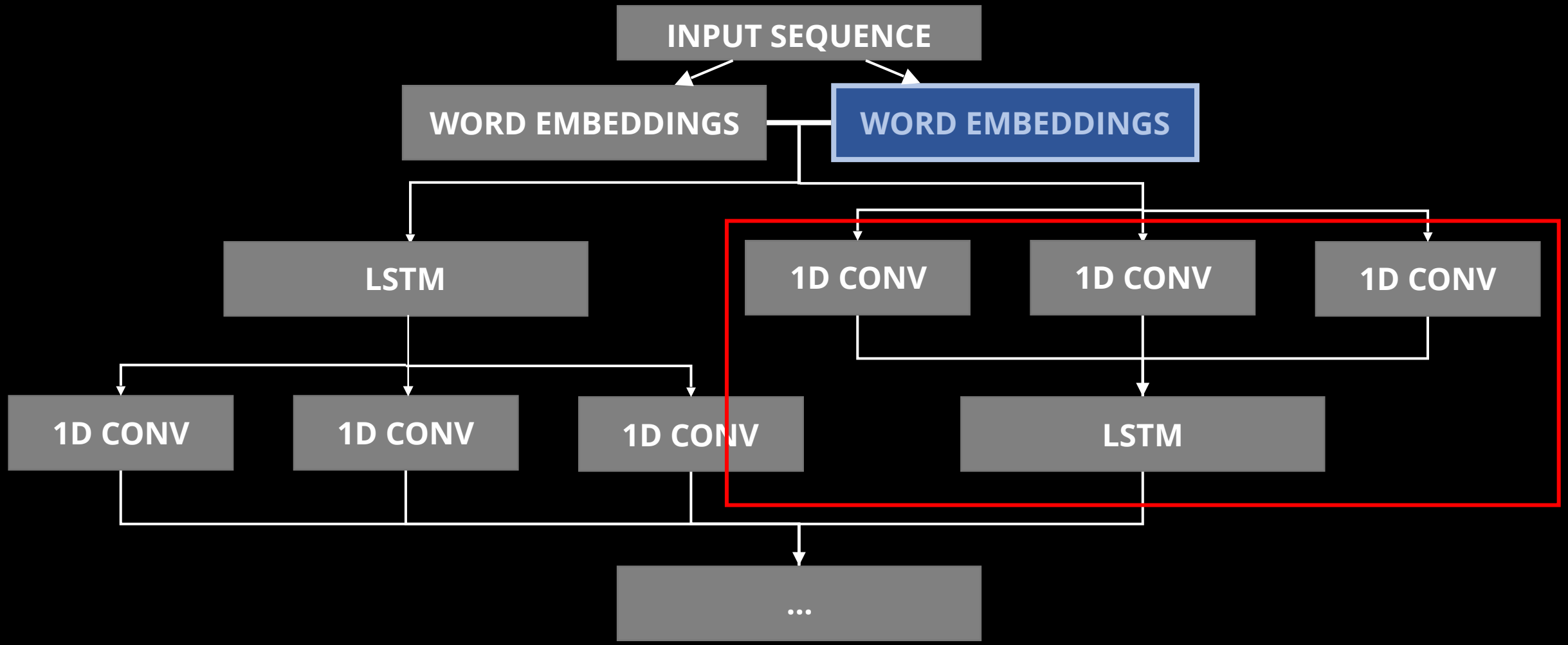
BalanceNet


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

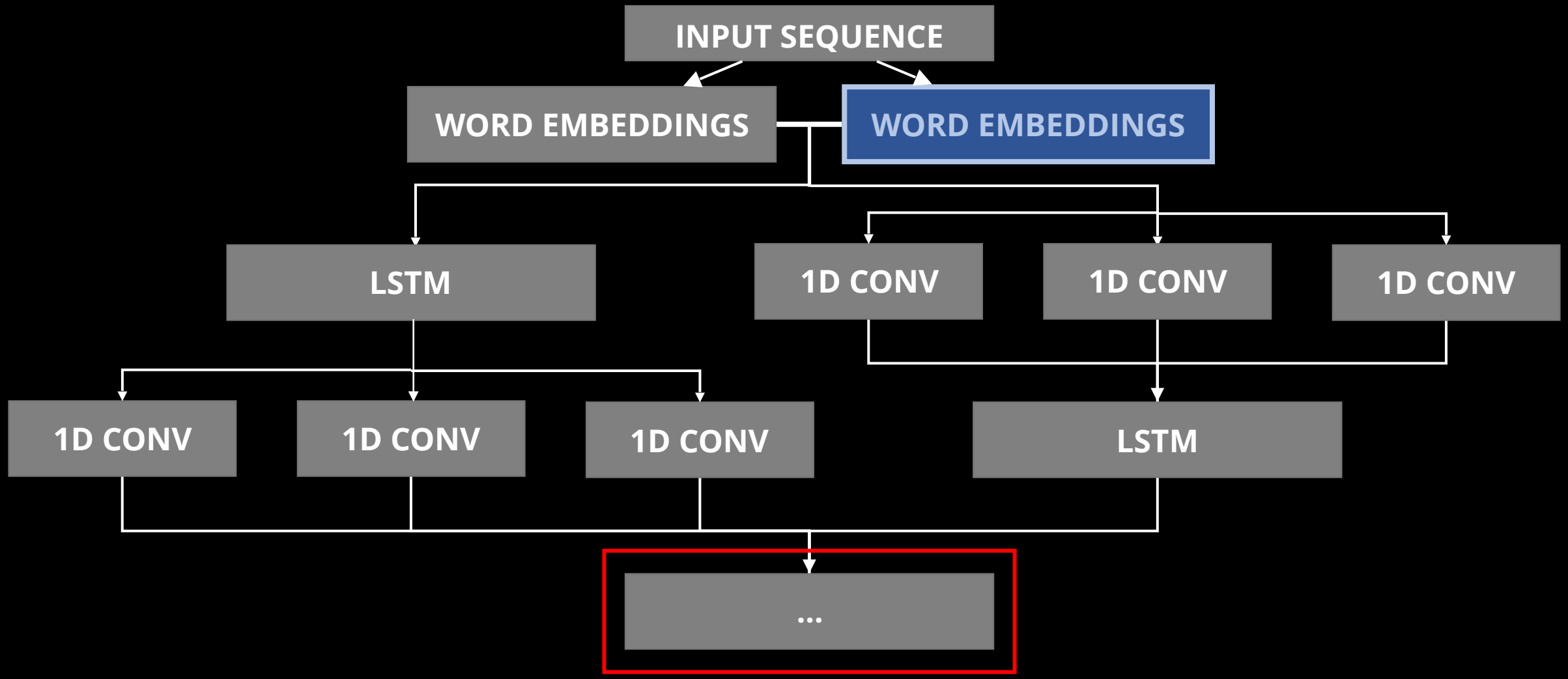
BalanceNet


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

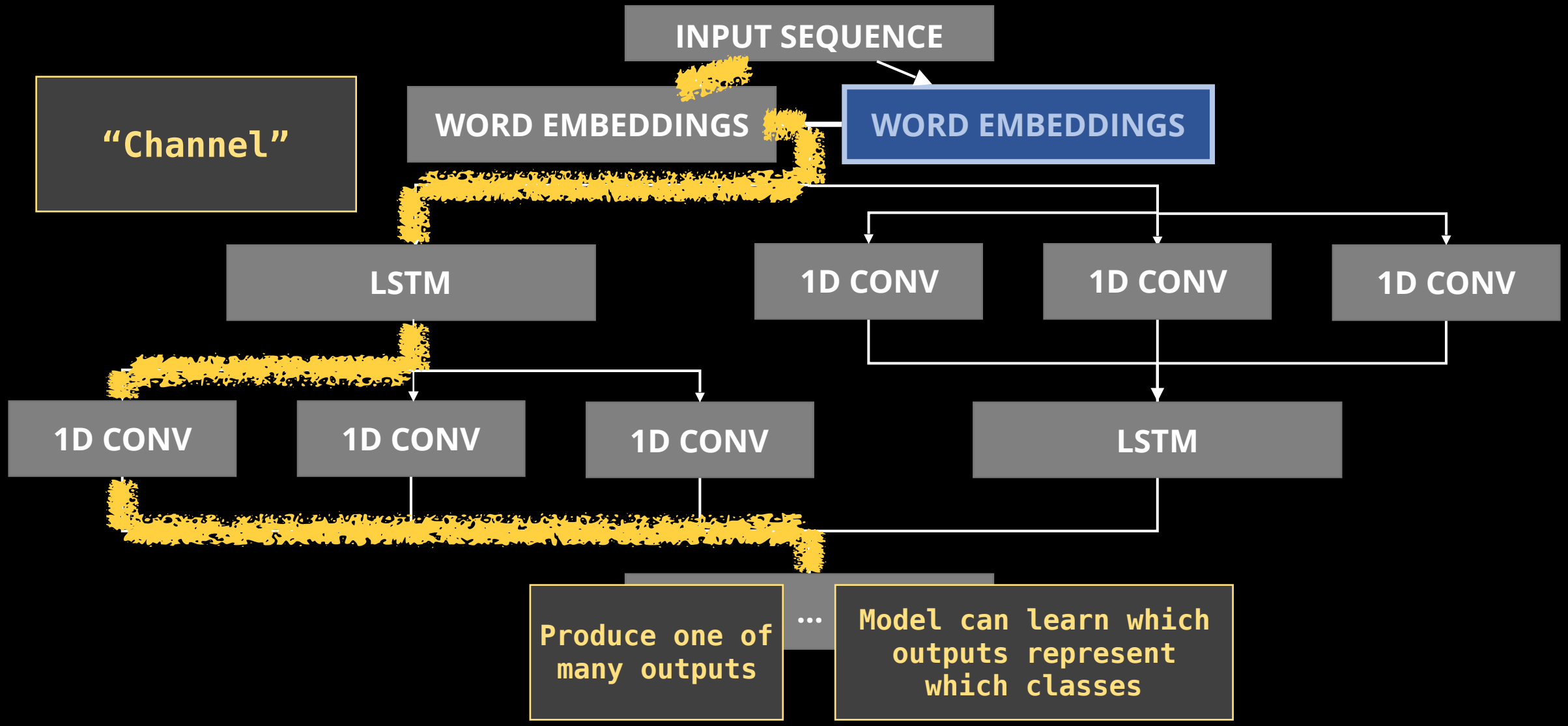
BalanceNet


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

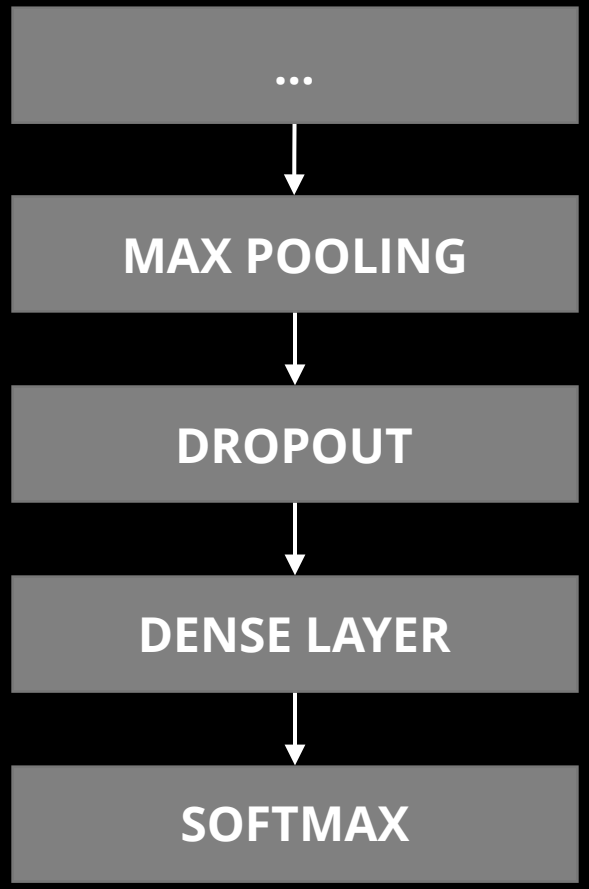
BalanceNet


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

BalanceNet


INPUT SEQUENCE
"Channel"
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...
Produce one of many outputs
Model can learn which outputs represent which classes

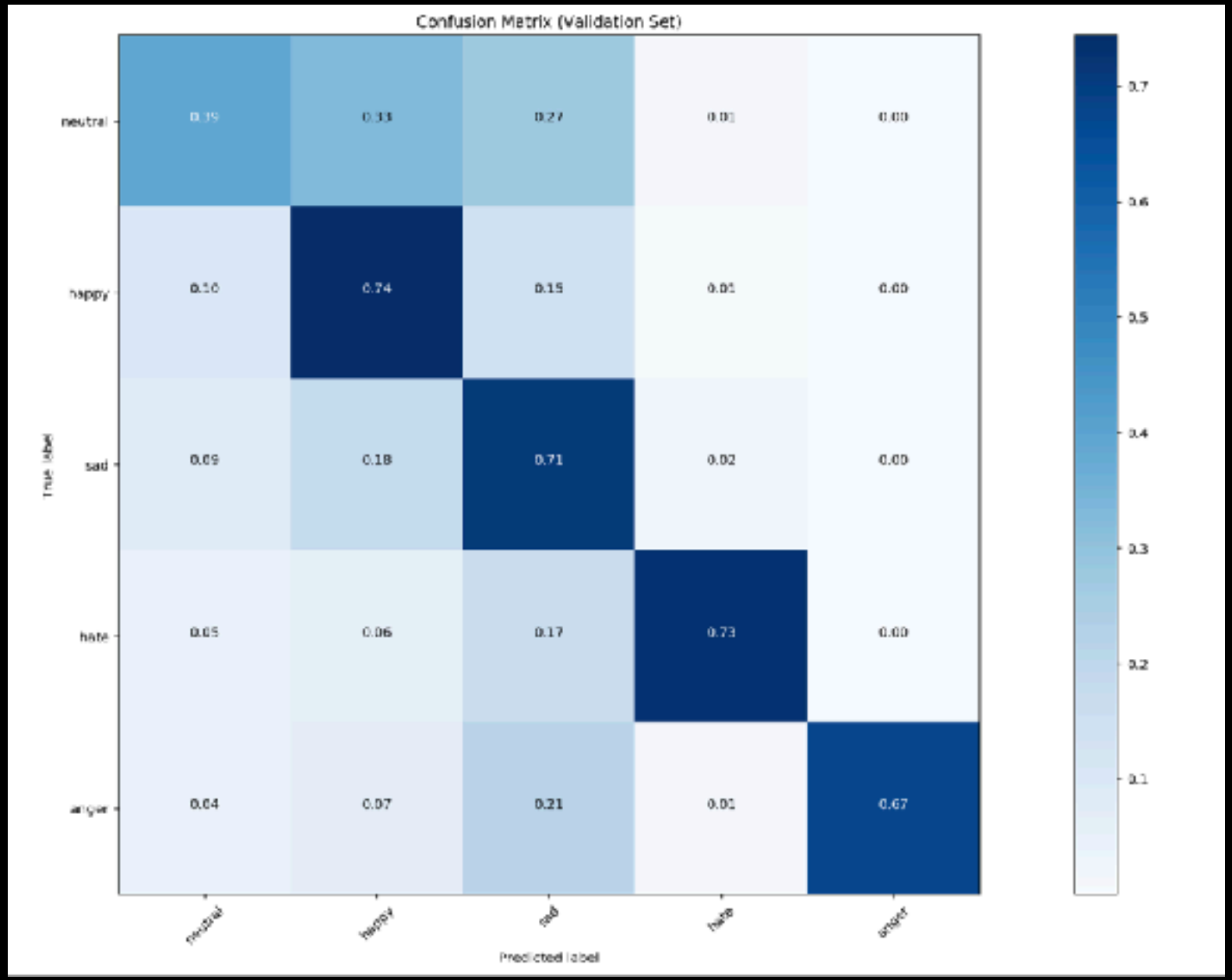
BalanceNet



Results

	precision	recall	f1-score	support
neutral	0.54	0.39	0.45	1939
happy	0.66	0.74	0.70	3237
sad	0.65	0.71	0.68	3192
hate	0.84	0.73	0.78	881
anger	0.95	0.67	0.79	208
avg / total	0.66	0.66	0.65	9457

Results


Confusion Matrix (Validation Set)
neutral
happy
sad
hate
angry
True label
0.39
0.33
0.27
0.01
0.00
0.10
0.74
0.15
0.01
0.00
0.09
0.18
0.71
0.02
0.00
0.05
0.06
0.17
0.73
0.00
0.04
0.07
0.21
0.01
0.67
neutral
happy
sad
hate
angry
Predicted label
0.7
0.6
0.5
0.4
0.3
0.2
0.1

Results (happy)

Haha look at all u dumbos cant read premium articles, im gonna becum premium nao to read their quality unbiased journalism
Prediction: happy

Wah gd deal. No much peepor there almost like whole area to urself. And if prepper still got boat for escape.
Hope i win 2mr toto, HUAT ARH!!!!!
Prediction: happy

I salute you for the bravery and sacrifice! A true hero indeed.
Prediction: happy

Results (sad)

Truly I said, we are so greatly divided in this world. Indeed money can do wonder, no one can deny it. Don't remind me, I know I'm still the loser.
Prediction: sad

what a nuisance fk. a proper clean and flat footpath,,now obstructed by sharedbikes..! which idiotic MP allowed this to happen?
Prediction: sad

Billions of money spent end up become the words 'unable to find out what's wrong with the system'
Prediction: sad

Results (hate)

Thought he sold his kidney to buy it; Instead, he bought a kidney then bought the car Filthy rich This is why we need communism
Prediction: hate

Sack the whole f[redacted]ing cabinet
Prediction: hate

For f[redacted]’s sake – news, stop calling it a machine. It was a guy in a f[redacted]ing box. Like that then glory hole macam innovative [redacted] machine.
Prediction: hate

Results (neutral)

ST really need to wake up.
Prediction: neutral

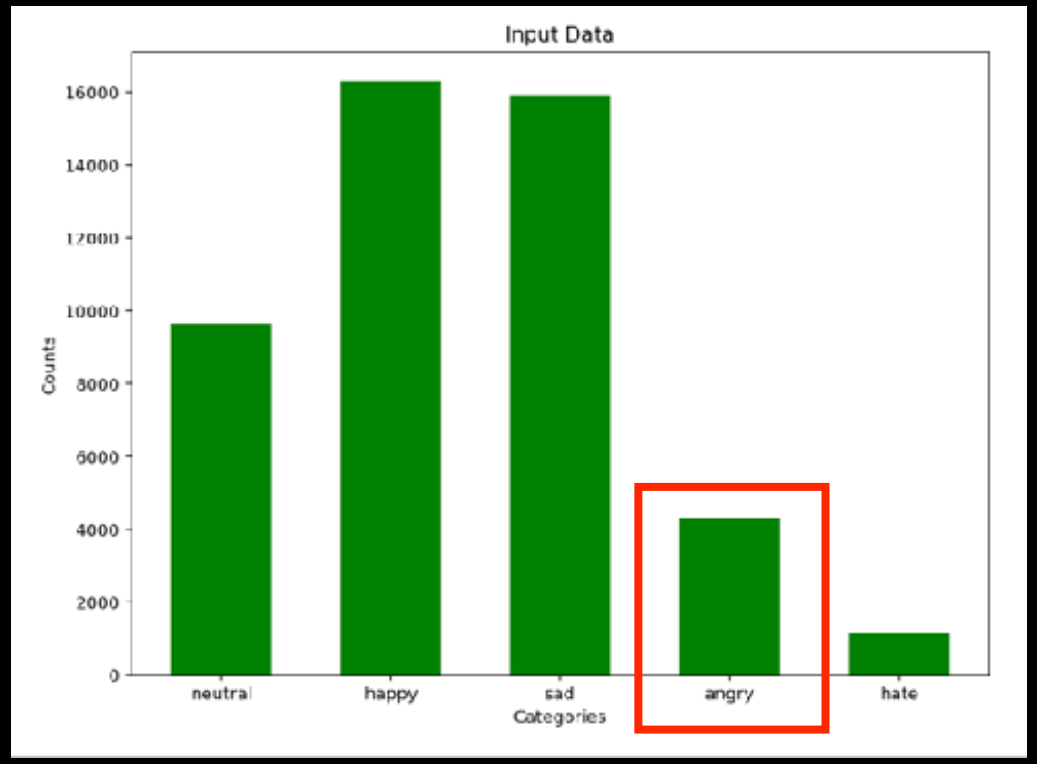
Dun need give birth ?? Come in PRC ok Liao
Prediction: neutral

Results (bonus?)

Somebody needs to water Tharman's head, hair needs to be grown there
Prediction: sad

What went wrong

- Distribution of training set



- Lower occurrences of “angry” tweets in training set results in less frequent/accurate predictions
- “hate” is more distinct so is easier to pick out

What went wrong

- What's up with the "neutral" class?
  - Not easy to set boundaries for "emotion-less" text
  - Hence the model picks up latent emotion
- Preprocessing could be better
  - Mistake: did not process the same way as Stanford (for their pre-trained GloVe vectors, which I used)

Future Plans

- Develop binary classifiers for emotion categories instead
- Develop sarcasm detector

Questions?

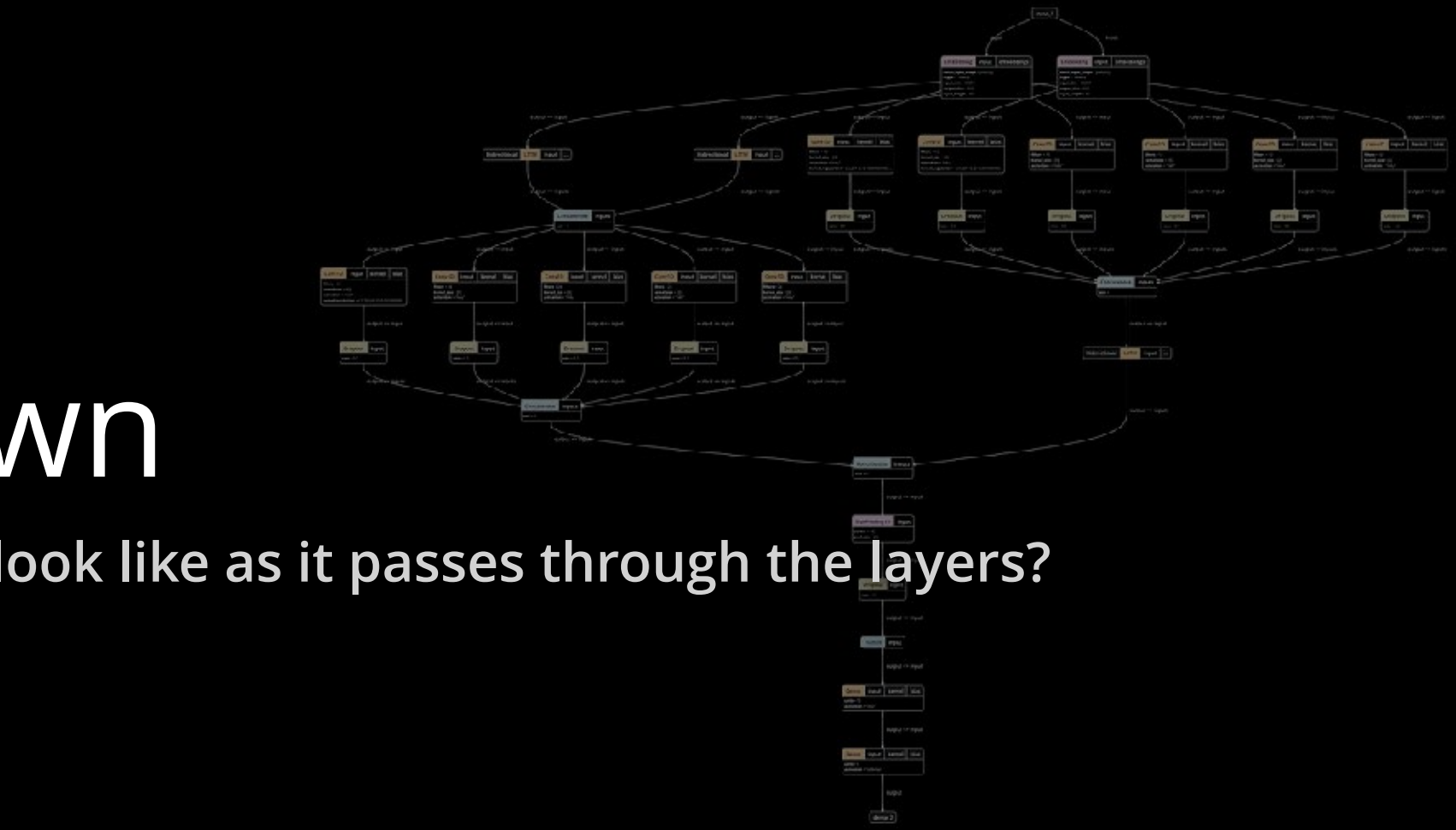
Thank you for listening!!

https://tlkh.design/

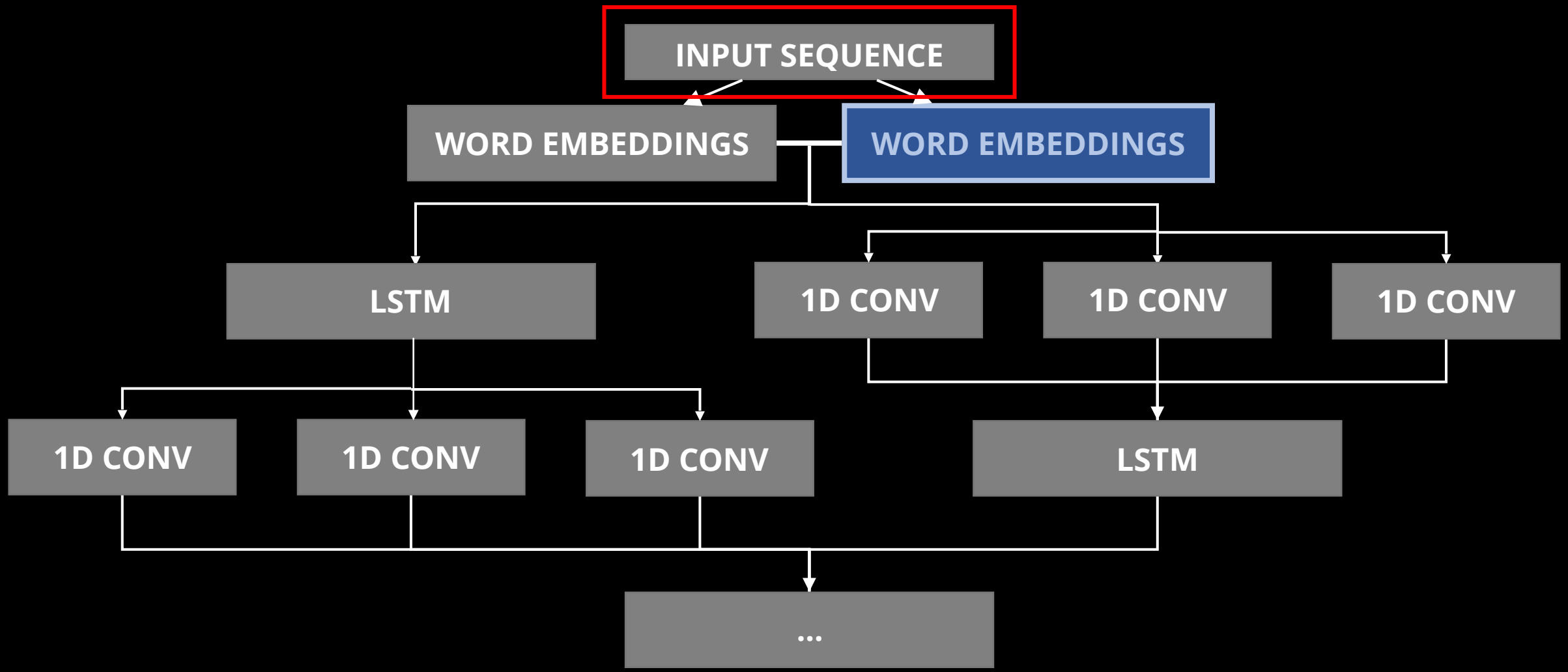
timothy_liu@mymail.sutd.edu.sg

Breakdown

What does the **`data`** look like as it passes through the layers?



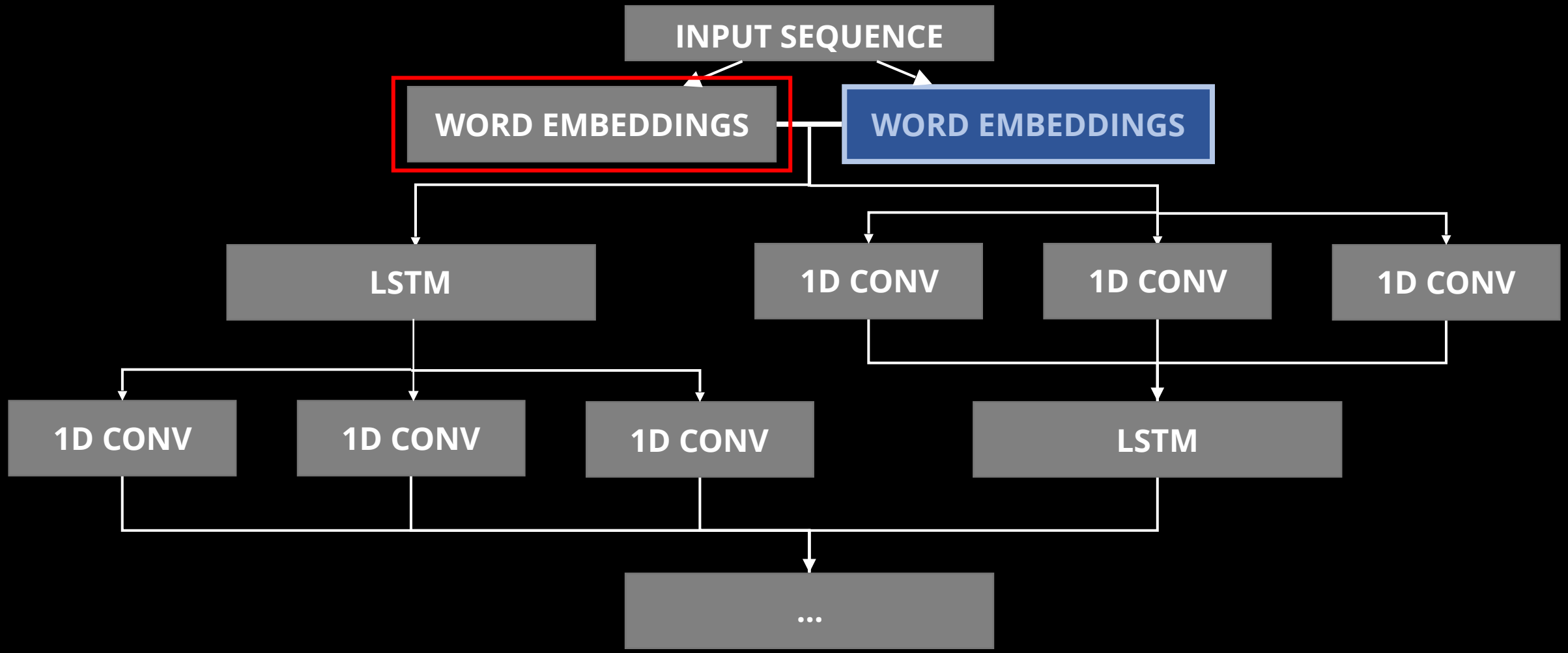
Breakdown


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

Breakdown

```
===== Layer name: input_1 =====
 Tensor:
 [[   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
      1 6097    5   12    3    6 4841    4  515 1476 1238    0    0    0
      0    0]]
 With shape: (1, 30)
=====
```

Breakdown


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

Breakdown

```
===== Layer name: input_1 =====
 Tensor:
 [[   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
      1 6097    5   12    3    6 4841    4  515 1476 1238    0    0    0
      0    0]]
 With shape: (1, 30)
=====
```

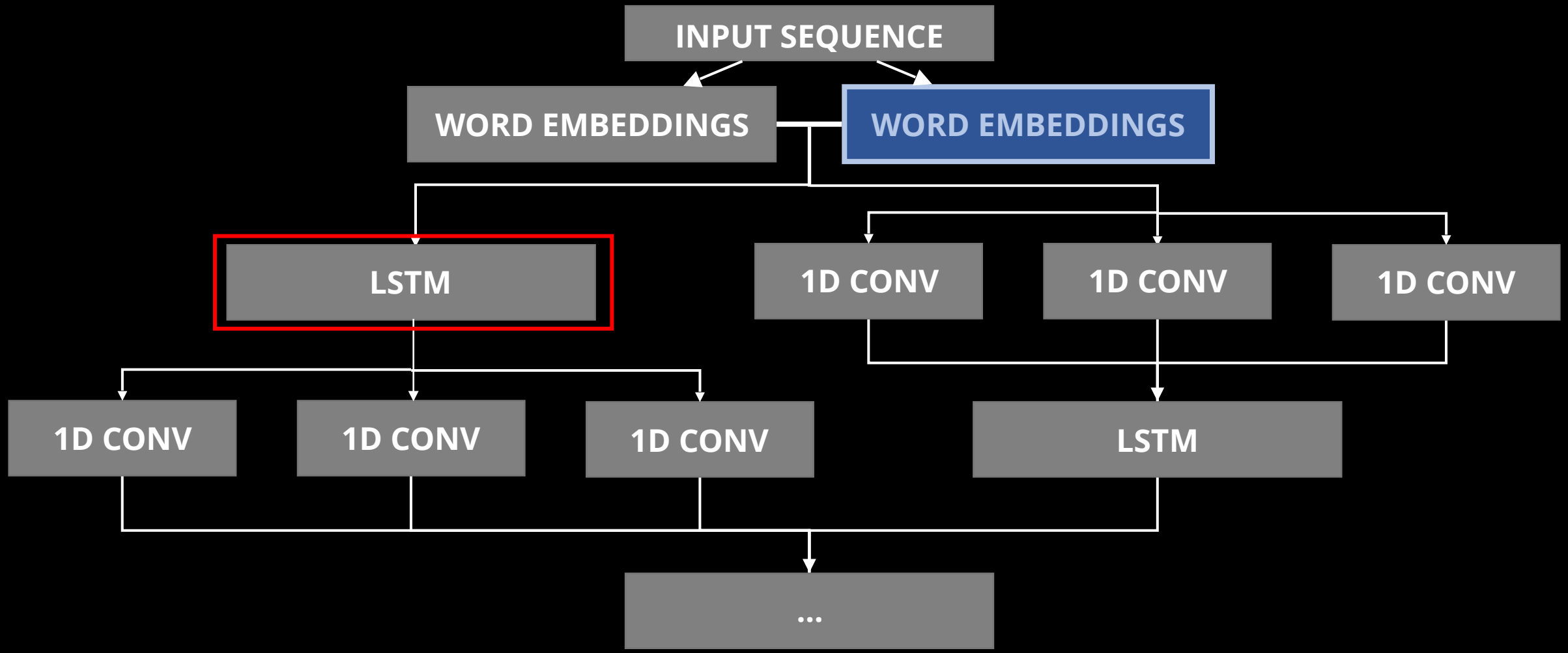
Breakdown

```
===== Layer name: input_1 =====
 Tensor:
 [[   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
      1 6097    5   12    3    6 200d    4  515 1476 1238    0    0    0
      0    0]]
 With shape: (1, 30)
=====
```

Breakdown

```
===== Layer name: embedding_1 =====
 Tensor:
 [[[0.42834136 0.31308484 0.17665115 ... 0.12682104 0.4330583  0.4869212 ]
  [0.42834136 0.31308484 0.17665115 ... 0.12682104 0.4330583  0.4869212 ]
  [0.42834136 0.31308484 0.17665115 ... 0.12682104 0.4330583  0.4869212 ]
  ...
  [0.42834136 0.31308484 0.17665115 ... 0.12682104 0.4330583  0.4869212 ]
  [0.42834136 0.31308484 0.17665115 ... 0.12682104 0.4330583  0.4869212 ]
  [0.42834136 0.31308484 0.17665115 ... 0.12682104 0.4330583  0.4869212 ]]]
 With shape: (1, 30, 200)
=====
```

Breakdown

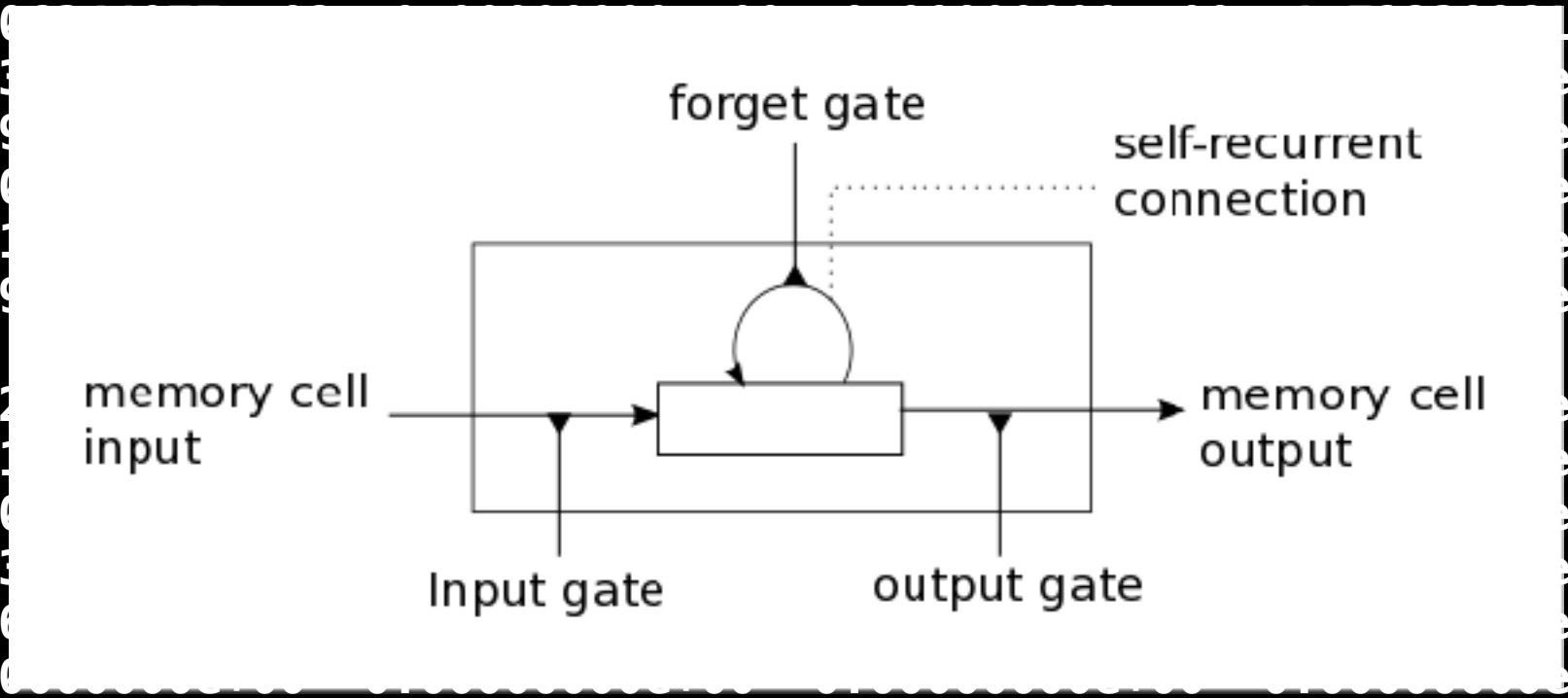

INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

Breakdown

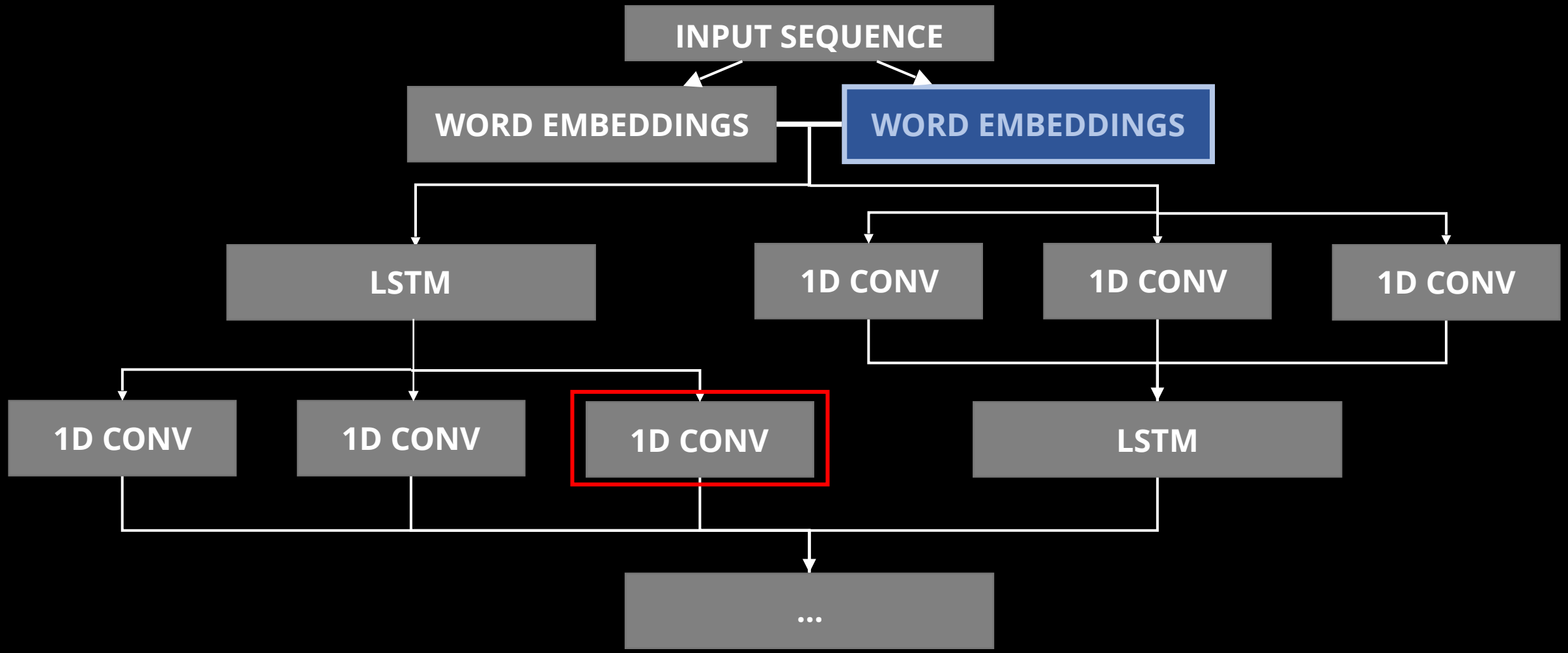
```
===== Layer name: bidirectional_1 =====
 Tensor:
 [[[ 8.06344077e-02  0.00000000e+00  0.00000000e+00  1.72238961e-01
    2.13927388e-01  0.00000000e+00 -1.32527430e-05 -2.44858768e-02
    2.49378700e-02 -9.10880626e-05  0.00000000e+00  1.62190315e-06]
  [ 9.80563536e-02  0.00000000e+00  0.00000000e+00  2.35380769e-01
    3.01944345e-01  0.00000000e+00 -2.48366414e-05 -2.44196467e-02
    2.89880596e-02 -1.35839160e-04  0.00000000e+00  3.31250180e-06]
  …
  [ 1.02114007e-01  0.00000000e+00 -1.75588634e-02  2.65173584e-01
    3.41420412e-01  0.00000000e+00  0.00000000e+00 -1.82297919e-02
    0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [ 1.03537120e-01  0.00000000e+00 -9.38578881e-03  2.70445615e-01
    3.56149286e-01  0.00000000e+00  0.00000000e+00 -1.20486589e-02
    0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]]
 With shape: (1, 30, 12)
=====
```

Breakdown

```
===== Layer name: bidirectional_1 =====
 Tensor:
 [[[ 8.0                                                                  e-01
    2.13                                                                  -02
    2.49                                                                  -06]
  [ 9.80                                                                  -01
    3.01                                                                  -02
    2.89                                                                  -06]
  …
  [ 1.02                                                                  -01
    3.41                                                                  -02
    0.00                                                                  +00]
  [ 1.03                                                                  -01
    3.56                                                                  -02
    0.00                                                                  +00]]]
 With shape: (1, 30, 12)
=====
```



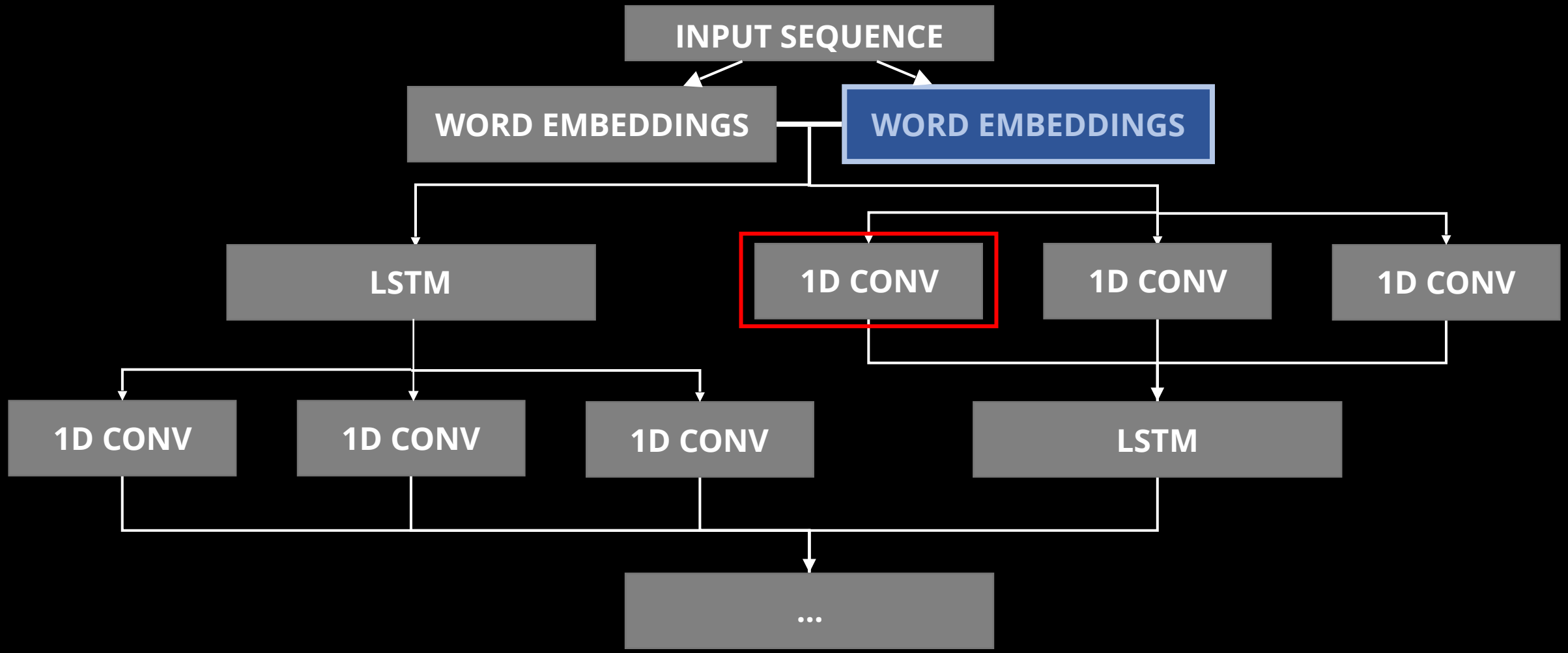
Breakdown


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

Breakdown

```
===== Layer name: conv1d_4 =====
 Tensor:
 [[[0.         0.         0.01488668 ... 0.11768723 0.         0.        ]
  [0.         0.         0.00893885 ... 0.12334982 0.         0.        ]
  [0.         0.         0.00739821 ... 0.12669875 0.         0.        ]
  ...
  [0.         0.         0.10555197 ... 0.1484237  0.08113632 0.2906651 ]
  [0.         0.         0.15426609 ... 0.10488616 0.         0.14050426]
  [0.         0.         0.06729852 ... 0.04035728 0.02787158 0.07842058]]]
 With shape: (1, 55, 24)
=====
```

Breakdown

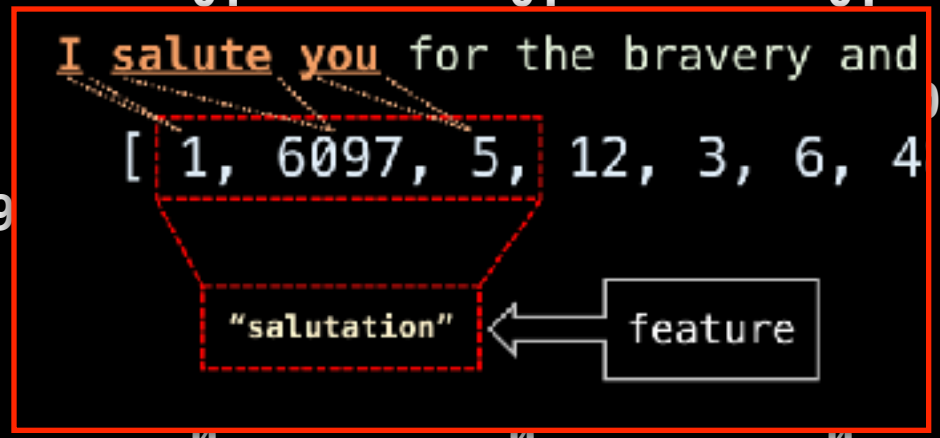

INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

Breakdown

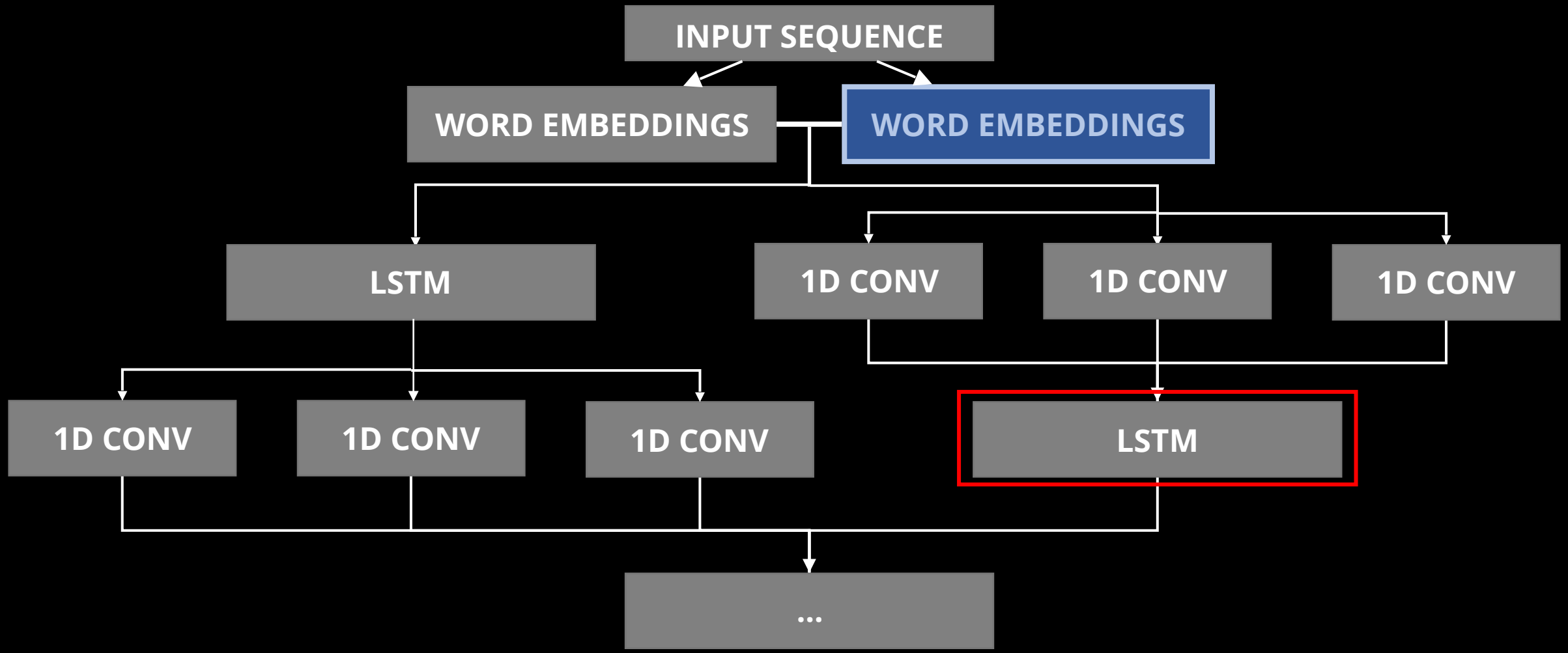
```
===== Layer name: conv1d_6 =====
 Tensor:
 [[[0.         0.         0.         0.         0.         0.
    0.         0.         0.         0.         0.         0.        ]
   ...
   [0.         0.         0.         0.         0.00264477 0.4029033
    0.34345472 0.         0.04512358 0.         0.         0.10371055]
   [0.         0.06970029 0.         0.         0.         0.
    0.10977992 0.         0.         0.         0.         0.        ]
   [0.         0.         0.         0.         0.         0.
    0.         0.         0.         0.         0.         0.        ]
   [0.         0.         0.         0.         0.         0.
    0.         0.         0.         0.         0.         0.        ]]]
 With shape: (1, 27, 12)
=====
```

Breakdown

```
===== Layer name: conv1d_6 =====
 Tensor:
 [[[0.         0.         0.         0.         0.         0.
    0.         0.         0.         0.         0.         0.        ]
   ...
    [0.         0.                                    0.264477 0.4029033
     0.34345472 0.                                    0.       0.10371055]
    [0.         0.069                                 0.       0.
     0.10977992 0.                                    0.       0.        ]
    [0.         0.                                    0.       0.
     0.         0.                                    0.       0.        ]
    [0.         0.         0.         0.         0.         0.
     0.         0.         0.         0.         0.         0.        ]]]
 With shape: (1, 27, 12)
=====
```



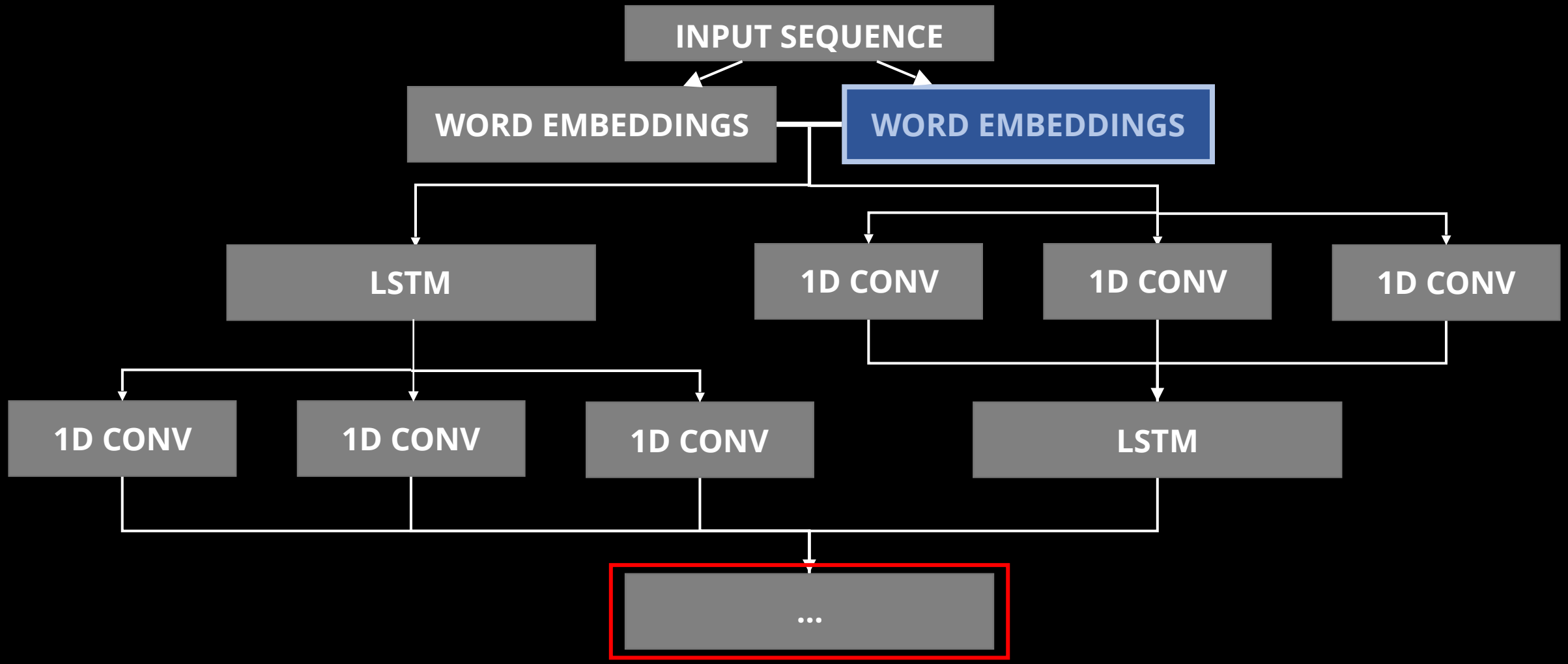
Breakdown


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

Breakdown

```
===== Layer name: bidirectional_3 =====
 Tensor:
 [[[ 2.21681572e-03 -1.28624942e-02  5.68648009e-03 ... -1.10270649e-01
   -8.20337683e-02 -1.02459006e-01]
  [ 1.03855331e-03 -2.48598233e-02  1.03258193e-02 ... -1.07269794e-01
   -8.28005373e-02 -1.07274570e-01]
  [-2.07568146e-03 -3.55149880e-02  1.36839077e-02 ... -1.03364743e-01
   -8.37882534e-02 -1.11438647e-01]
  ...
  [ 1.92181729e-02 -1.31640313e-02  6.91433847e-02 ... -4.64378782e-02
   -2.86338408e-03  1.77442562e-03]
  [ 2.66722459e-02 -3.44170779e-02  6.45167008e-02 ... -3.46677154e-02
   -8.64953792e-04 -9.22709223e-05]
  [ 2.77910624e-02 -5.21486029e-02  6.20827414e-02 ... -1.94451436e-02
    1.80956413e-04 -8.93047196e-04]]]
 With shape: (1, 168, 24)
=====
```

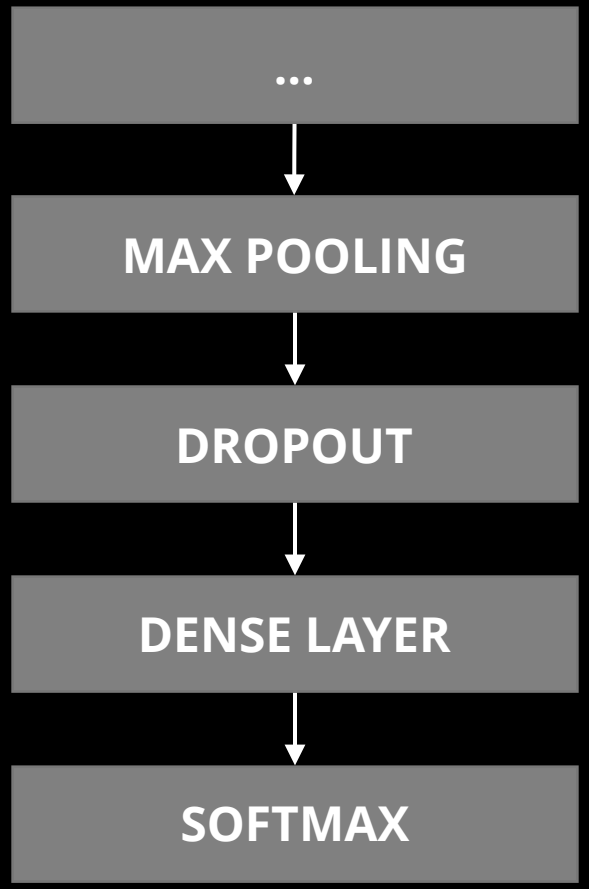
Breakdown


INPUT SEQUENCE
WORD EMBEDDINGS
WORD EMBEDDINGS
LSTM
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
1D CONV
LSTM
...

Breakdown

```
===== Layer name: concatenate_4 =====
 Tensor:
 [[[ 0.0000000e+00  0.0000000e+00  1.4886685e-02 ...  1.1768723e-01
     0.0000000e+00  0.0000000e+00]
  [ 0.0000000e+00  0.0000000e+00  8.9388452e-03 ...  1.2334982e-01
     0.0000000e+00  0.0000000e+00]
  [ 0.0000000e+00  0.0000000e+00  7.3982142e-03 ...  1.2669875e-01
     0.0000000e+00  0.0000000e+00]
  ...
  [ 1.9218173e-02 -1.3164031e-02  6.9143385e-02 ... -4.6437878e-02
   -2.8633841e-03  1.7744256e-03]
  [ 2.6672246e-02 -3.4417078e-02  6.4516701e-02 ... -3.4667715e-02
   -8.6495379e-04 -9.2270922e-05]
  [ 2.7791062e-02 -5.2148603e-02  6.2082741e-02 ... -1.9445144e-02
     1.8095641e-04 -8.9304720e-04]]]
 With shape: (1, 449, 24)
=====
```

Breakdown



Breakdown

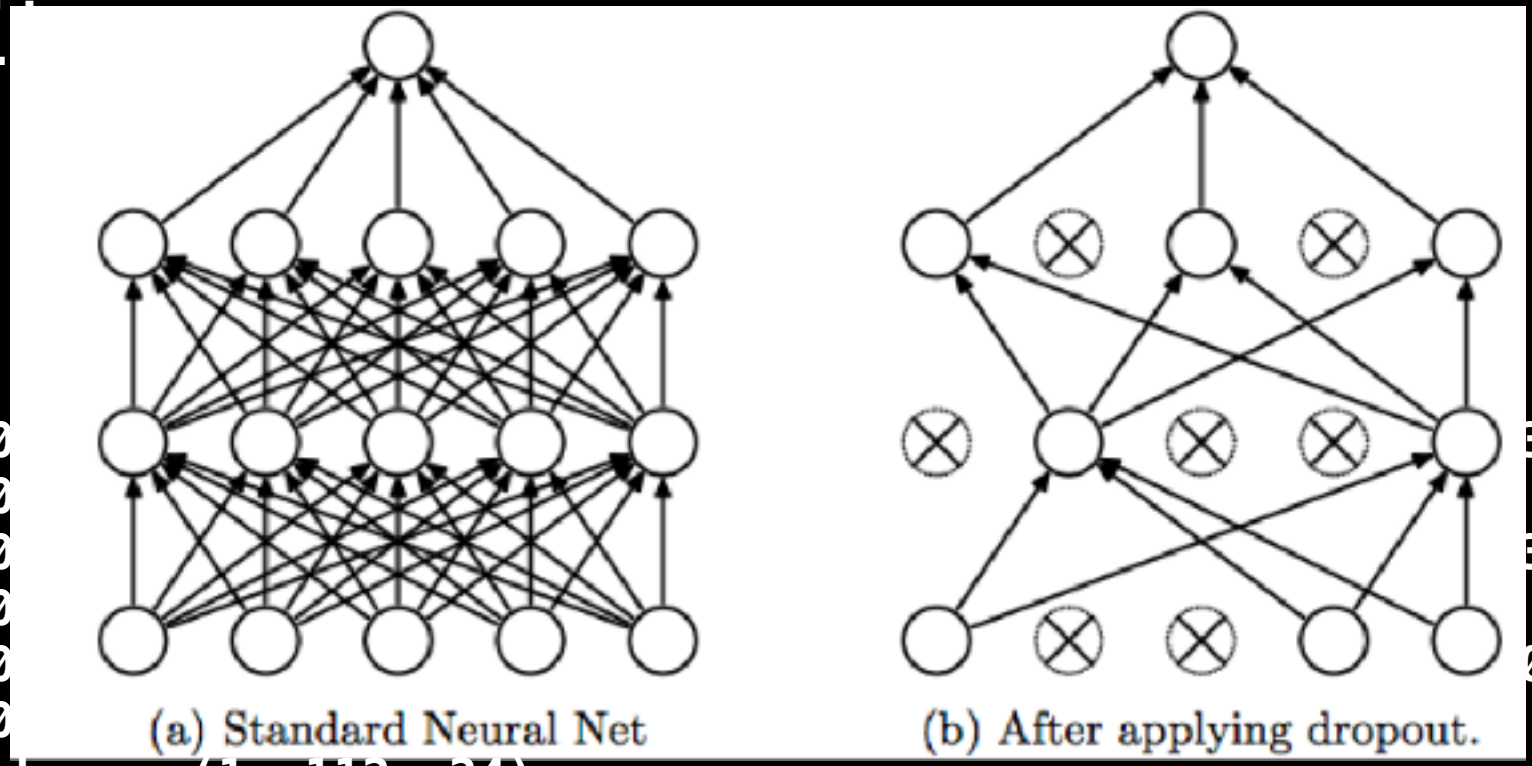
```
===== Layer name: max_pooling1d_1 =====
 Tensor:
 [[[ 0.          0.          0.01488668 ...  0.12903301  0.
    0.        ]
  [ 0.          0.          0.01099134 ...  0.13447145  0.
    0.        ]
  [ 0.          0.          0.01290299 ...  0.13462676  0.
    0.        ]
  ...
  [-0.05108929  0.02756429  0.19028117 ...  0.13576919 -0.03558763
    0.0127592 ]
  [ 0.02745983  0.12298487  0.21508092 ...  0.09027248 -0.0352531
    0.06895266]
  [ 0.02667225  0.04184575  0.10012361 ... -0.03466772 -0.00086495
    0.00427817]]]
 With shape: (1, 112, 24)
=====
```

Breakdown

```
===== Layer name: dropout_12 =====
 Tensor:
 [[[ 0.          0.          0.01488668 ...  0.12903301  0.
    0.        ]
  [ 0.          0.          0.01099134 ...  0.13447145  0.
    0.        ]
  [ 0.          0.          0.01290299 ...  0.13462676  0.
    0.        ]
  ...
  [-0.05108929  0.02756429  0.19028117 ...  0.13576919 -0.03558763
    0.0127592 ]
  [ 0.02745983  0.12298487  0.21508092 ...  0.09027248 -0.0352531
    0.06895266]
  [ 0.02667225  0.04184575  0.10012361 ... -0.03466772 -0.00086495
    0.00427817]]]
 With shape: (1, 112, 24)
=====
```

Breakdown

```
===== Layer name: dropout_12 =====
 Tensor:
 [[[ 0.
    0.
  [ 0.
    0.
  [ 0.
    0.
  ...
  [-0.0                                                    3558763
    0.0
  [ 0.0                                                    352531
    0.0
  [ 0.0                                                    0086495
    0.0
 With shape: (1, 112, 24)
=====
```



(a) Standard Neural Net

(b) After applying dropout.

Breakdown

```
===== Layer name: dense_1 =====
 Tensor:
 [[1.6699548  0.40102258 0.         0.         0.62673044 0.07761759
  0.         4.3026257  0.         0.16274107 1.9700971  0.
  0.03963258 0.83466715 0.42836624 0.         0.         0.
  2.36443    4.663144   0.         0.         0.         0.0437276
  0.         0.        ]]
 With shape: (1, 26)
=====

===== Layer name: dense_2 =====
 Tensor:
 [[0.06425636 0.8729007  0.05157411 0.00984968 0.00141912]]
 With shape: (1, 5)
=====
```

Breakdown

```
===== Layer name: de
 Tensor:
 [[1.6699548  0.401                                  0.62673044 0.07761759
  0.         4.3026                                 1.9700971  0.
  0.03963258 0.8346                                 0.         0.
  2.36443    4.6631                                 0.         0.0437276
  0.         0.
 With shape: (1, 26
=====

===== Layer name: de
 Tensor:
 [[0.06425636 0.872                                  0.00141912]]
 With shape: (1, 5)
=====
```