Overfitting and Underfitting

June 16, 2019

1 Overfitting and underfitting

Overfit When the model adapts to the particularities of the training set. We want to generalize it for better performance in unknown data.

It can happen when: - we train the model for too long (the validation accuracy hits a peak and starts decreasing) - the model has too much freedom (too many layers or hidden units)

Solutions: 1. Train on large datasets 2. Use regularization 3. Find the best training duration

Underfit When there is still room for improvement on test data.

It can happen when: - the model is not powerful enough - it is overregularized - was not trained enough

[1]:
```
from __future__ import absolute_import, division, print_function

import tensorflow as tf
from tensorflow import keras

import numpy as np
import matplotlib.pyplot as plt

print(tf.__version__)
```

```
1.13.1
```

1.1 Preprocessing

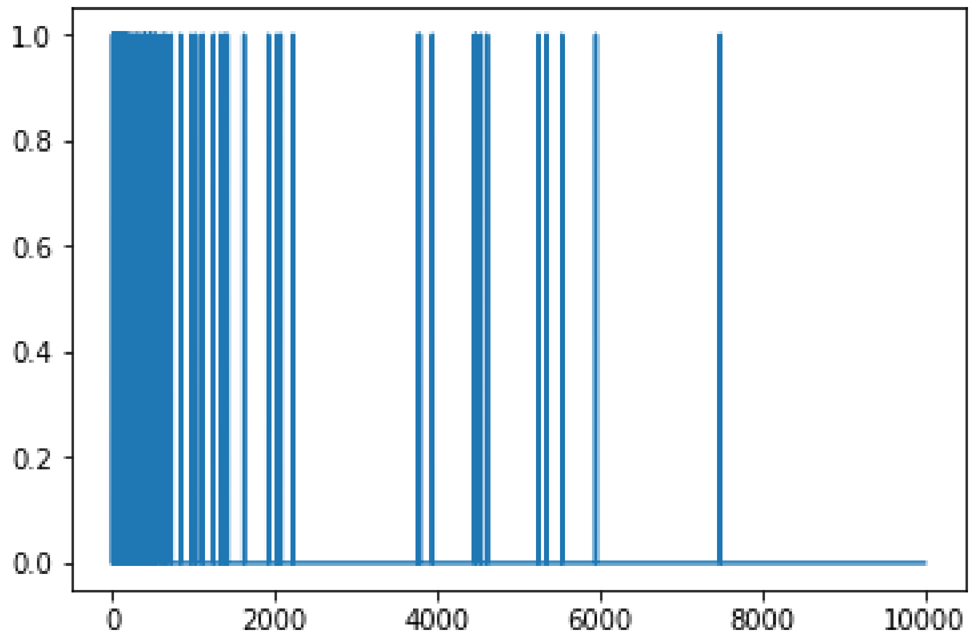
We will use the IMDB review dataset with multi-hot encoding ([3, 5] => [000101000. . . 00]) instead of embedding. The model will quickly overfit.

[2]:
```
NUM_WORDS = 10000

(train_data, train_labels), (test_data, test_labels) = keras.datasets.imdb.
 ↪load_data(num_words=NUM_WORDS)

def multi_hot_sequences(sequences, dimension):
    # Create an all-zero matrix of shape (len(sequences), dimension)
    results = np.zeros((len(sequences), dimension))
    for i, word_indices in enumerate(sequences):
        results[i, word_indices] = 1.0  # set specific indices of results[i] to␣
↪1s
    return results


train_data = multi_hot_sequences(train_data, dimension=NUM_WORDS)
test_data = multi_hot_sequences(test_data, dimension=NUM_WORDS)
```

Since the words are sorted byt frequence, we expect more 1's near index 0

[3]:
```
plt.plot(train_data[0])
```

[3]:
```
[<matplotlib.lines.Line2D at 0x2d2985021d0>]
```



1.2 About overfitting

The simplest way to prevent overfitting is to reduce the size (**capacity** = number of learnable parameters = number of layers and units) so that it will focus on more important patterns with more predictive power.

To find the best architecture, the best is to start with just a few layers and units and then increase them until validation loss stops improving.

Baseline model

[4]:
```
baseline_model = keras.Sequential([
    # `input_shape` is only required here so that `.summary` works.
    keras.layers.Dense(16, activation=tf.nn.relu, input_shape=(NUM_WORDS,)),
    keras.layers.Dense(16, activation=tf.nn.relu),
    keras.layers.Dense(1, activation=tf.nn.sigmoid)
])

baseline_model.compile(optimizer='adam',
                       loss='binary_crossentropy',
                       metrics=['accuracy', 'binary_crossentropy'])

baseline_model.summary()
```

```
WARNING:tensorflow:From C:\Users\Pedro\AppData\Roaming\Python\Python37\site-
packages\tensorflow\python\ops\resource_variable_ops.py:435: colocate_with (from
tensorflow.python.framework.ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Colocations handled automatically by placer.
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 16)                160016
_________________________________________________________________
dense_1 (Dense)              (None, 16)                272
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 17
=================================================================
Total params: 160,305
Trainable params: 160,305
Non-trainable params: 0
_________________________________________________________________
```

[5]:
```
baseline_history = baseline_model.fit(train_data,
                                      train_labels,
                                      epochs=20,
                                      batch_size=512,
                                      validation_data=(test_data, test_labels),
                                      verbose=2)
```

```
Train on 25000 samples, validate on 25000 samples
WARNING:tensorflow:From C:\Users\Pedro\AppData\Roaming\Python\Python37\site-
packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from
tensorflow.python.ops.math_ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Use tf.cast instead.
Epoch 1/20
 - 19s - loss: 0.4814 - acc: 0.8046 - binary_crossentropy: 0.4814 - val_loss:
0.3301 - val_acc: 0.8766 - val_binary_crossentropy: 0.3301
Epoch 2/20
```

```
 - 15s - loss: 0.2442 - acc: 0.9126 - binary_crossentropy: 0.2442 - val_loss:
0.2835 - val_acc: 0.8885 - val_binary_crossentropy: 0.2835
Epoch 3/20
 - 11s - loss: 0.1777 - acc: 0.9376 - binary_crossentropy: 0.1777 - val_loss:
0.2952 - val_acc: 0.8828 - val_binary_crossentropy: 0.2952
Epoch 4/20
 - 8s - loss: 0.1449 - acc: 0.9506 - binary_crossentropy: 0.1449 - val_loss:
0.3374 - val_acc: 0.8708 - val_binary_crossentropy: 0.3374
Epoch 5/20
 - 8s - loss: 0.1193 - acc: 0.9604 - binary_crossentropy: 0.1193 - val_loss:
0.3429 - val_acc: 0.8736 - val_binary_crossentropy: 0.3429
Epoch 6/20
 - 7s - loss: 0.0981 - acc: 0.9695 - binary_crossentropy: 0.0981 - val_loss:
0.3738 - val_acc: 0.8693 - val_binary_crossentropy: 0.3738
Epoch 7/20
 - 7s - loss: 0.0817 - acc: 0.9753 - binary_crossentropy: 0.0817 - val_loss:
0.4114 - val_acc: 0.8656 - val_binary_crossentropy: 0.4114
Epoch 8/20
 - 7s - loss: 0.0699 - acc: 0.9806 - binary_crossentropy: 0.0699 - val_loss:
0.4512 - val_acc: 0.8621 - val_binary_crossentropy: 0.4512
Epoch 9/20
 - 7s - loss: 0.0563 - acc: 0.9858 - binary_crossentropy: 0.0563 - val_loss:
0.4873 - val_acc: 0.8606 - val_binary_crossentropy: 0.4873
Epoch 10/20
 - 7s - loss: 0.0475 - acc: 0.9889 - binary_crossentropy: 0.0475 - val_loss:
0.5306 - val_acc: 0.8577 - val_binary_crossentropy: 0.5306
Epoch 11/20
 - 12s - loss: 0.0383 - acc: 0.9928 - binary_crossentropy: 0.0383 - val_loss:
0.5761 - val_acc: 0.8559 - val_binary_crossentropy: 0.5761
Epoch 12/20
 - 10s - loss: 0.0308 - acc: 0.9946 - binary_crossentropy: 0.0308 - val_loss:
0.6153 - val_acc: 0.8537 - val_binary_crossentropy: 0.6153
Epoch 13/20
 - 7s - loss: 0.0255 - acc: 0.9963 - binary_crossentropy: 0.0255 - val_loss:
0.6728 - val_acc: 0.8514 - val_binary_crossentropy: 0.6728
Epoch 14/20
 - 6s - loss: 0.0199 - acc: 0.9978 - binary_crossentropy: 0.0199 - val_loss:
0.7107 - val_acc: 0.8506 - val_binary_crossentropy: 0.7107
Epoch 15/20
 - 6s - loss: 0.0157 - acc: 0.9984 - binary_crossentropy: 0.0157 - val_loss:
0.7351 - val_acc: 0.8498 - val_binary_crossentropy: 0.7351
Epoch 16/20
 - 7s - loss: 0.0125 - acc: 0.9991 - binary_crossentropy: 0.0125 - val_loss:
0.7775 - val_acc: 0.8498 - val_binary_crossentropy: 0.7775
Epoch 17/20
 - 7s - loss: 0.0103 - acc: 0.9994 - binary_crossentropy: 0.0103 - val_loss:
0.8070 - val_acc: 0.8488 - val_binary_crossentropy: 0.8070
Epoch 18/20
```

```
 - 6s - loss: 0.0084 - acc: 0.9996 - binary_crossentropy: 0.0084 - val_loss:
0.8404 - val_acc: 0.8485 - val_binary_crossentropy: 0.8404
Epoch 19/20
 - 6s - loss: 0.0069 - acc: 0.9996 - binary_crossentropy: 0.0069 - val_loss:
0.8704 - val_acc: 0.8485 - val_binary_crossentropy: 0.8704
Epoch 20/20
 - 6s - loss: 0.0057 - acc: 0.9997 - binary_crossentropy: 0.0057 - val_loss:
0.8976 - val_acc: 0.8476 - val_binary_crossentropy: 0.8976
```

Smaller model

[6]:
```
smaller_model = keras.Sequential([
    keras.layers.Dense(4, activation=tf.nn.relu, input_shape=(NUM_WORDS,)),
    keras.layers.Dense(4, activation=tf.nn.relu),
    keras.layers.Dense(1, activation=tf.nn.sigmoid)
])

smaller_model.compile(optimizer='adam',
                loss='binary_crossentropy',
                metrics=['accuracy', 'binary_crossentropy'])

smaller_model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 4)                 40004
_________________________________________________________________
dense_4 (Dense)              (None, 4)                 20
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 5
=================================================================
Total params: 40,029
Trainable params: 40,029
Non-trainable params: 0
_________________________________________________________________
```

[7]:
```
smaller_history = smaller_model.fit(train_data,
                                    train_labels,
                                    epochs=20,
                                    batch_size=512,
                                    validation_data=(test_data, test_labels),
                                    verbose=2)
```

```
Train on 25000 samples, validate on 25000 samples
Epoch 1/20
 - 6s - loss: 0.6613 - acc: 0.5813 - binary_crossentropy: 0.6613 - val_loss:
0.6218 - val_acc: 0.6428 - val_binary_crossentropy: 0.6218
```

```
Epoch 2/20
 - 6s - loss: 0.5774 - acc: 0.7418 - binary_crossentropy: 0.5774 - val_loss:
0.5550 - val_acc: 0.7581 - val_binary_crossentropy: 0.5550
Epoch 3/20
 - 6s - loss: 0.5126 - acc: 0.8280 - binary_crossentropy: 0.5126 - val_loss:
0.5091 - val_acc: 0.8053 - val_binary_crossentropy: 0.5091
Epoch 4/20
 - 6s - loss: 0.4657 - acc: 0.8676 - binary_crossentropy: 0.4657 - val_loss:
0.4767 - val_acc: 0.8368 - val_binary_crossentropy: 0.4767
Epoch 5/20
 - 6s - loss: 0.4300 - acc: 0.8933 - binary_crossentropy: 0.4300 - val_loss:
0.4560 - val_acc: 0.8439 - val_binary_crossentropy: 0.4560
Epoch 6/20
 - 6s - loss: 0.4015 - acc: 0.9086 - binary_crossentropy: 0.4015 - val_loss:
0.4361 - val_acc: 0.8650 - val_binary_crossentropy: 0.4361
Epoch 7/20
 - 6s - loss: 0.3772 - acc: 0.9203 - binary_crossentropy: 0.3772 - val_loss:
0.4238 - val_acc: 0.8692 - val_binary_crossentropy: 0.4238
Epoch 8/20
 - 6s - loss: 0.3563 - acc: 0.9313 - binary_crossentropy: 0.3563 - val_loss:
0.4176 - val_acc: 0.8661 - val_binary_crossentropy: 0.4176
Epoch 9/20
 - 5s - loss: 0.3376 - acc: 0.9391 - binary_crossentropy: 0.3376 - val_loss:
0.4125 - val_acc: 0.8662 - val_binary_crossentropy: 0.4125
Epoch 10/20
 - 5s - loss: 0.3200 - acc: 0.9460 - binary_crossentropy: 0.3200 - val_loss:
0.4012 - val_acc: 0.8716 - val_binary_crossentropy: 0.4012
Epoch 11/20
 - 6s - loss: 0.2960 - acc: 0.9521 - binary_crossentropy: 0.2960 - val_loss:
0.3901 - val_acc: 0.8725 - val_binary_crossentropy: 0.3901
Epoch 12/20
 - 6s - loss: 0.2582 - acc: 0.9584 - binary_crossentropy: 0.2582 - val_loss:
0.3659 - val_acc: 0.8762 - val_binary_crossentropy: 0.3659
Epoch 13/20
 - 6s - loss: 0.2134 - acc: 0.9636 - binary_crossentropy: 0.2134 - val_loss:
0.3525 - val_acc: 0.8750 - val_binary_crossentropy: 0.3525
Epoch 14/20
 - 6s - loss: 0.1540 - acc: 0.9683 - binary_crossentropy: 0.1540 - val_loss:
0.3330 - val_acc: 0.8749 - val_binary_crossentropy: 0.3330
Epoch 15/20
 - 6s - loss: 0.1172 - acc: 0.9713 - binary_crossentropy: 0.1172 - val_loss:
0.3657 - val_acc: 0.8711 - val_binary_crossentropy: 0.3657
Epoch 16/20
 - 6s - loss: 0.1006 - acc: 0.9757 - binary_crossentropy: 0.1006 - val_loss:
0.3760 - val_acc: 0.8706 - val_binary_crossentropy: 0.3760
Epoch 17/20
 - 6s - loss: 0.0882 - acc: 0.9801 - binary_crossentropy: 0.0882 - val_loss:
0.3924 - val_acc: 0.8690 - val_binary_crossentropy: 0.3924
```

```
Epoch 18/20
 - 6s - loss: 0.0788 - acc: 0.9825 - binary_crossentropy: 0.0788 - val_loss:
0.4186 - val_acc: 0.8691 - val_binary_crossentropy: 0.4186
Epoch 19/20
 - 6s - loss: 0.0709 - acc: 0.9850 - binary_crossentropy: 0.0709 - val_loss:
0.4316 - val_acc: 0.8670 - val_binary_crossentropy: 0.4316
Epoch 20/20
 - 6s - loss: 0.0640 - acc: 0.9870 - binary_crossentropy: 0.0640 - val_loss:
0.4495 - val_acc: 0.8664 - val_binary_crossentropy: 0.4495
```

Bigger model

[8]:
```
bigger_model = keras.models.Sequential([
    keras.layers.Dense(512, activation=tf.nn.relu, input_shape=(NUM_WORDS,)),
    keras.layers.Dense(512, activation=tf.nn.relu),
    keras.layers.Dense(1, activation=tf.nn.sigmoid)
])

bigger_model.compile(optimizer='adam',
                     loss='binary_crossentropy',
                     metrics=['accuracy','binary_crossentropy'])

bigger_model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_6 (Dense)              (None, 512)               5120512
_________________________________________________________________
dense_7 (Dense)              (None, 512)               262656
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 513
=================================================================
Total params: 5,383,681
Trainable params: 5,383,681
Non-trainable params: 0
_________________________________________________________________
```

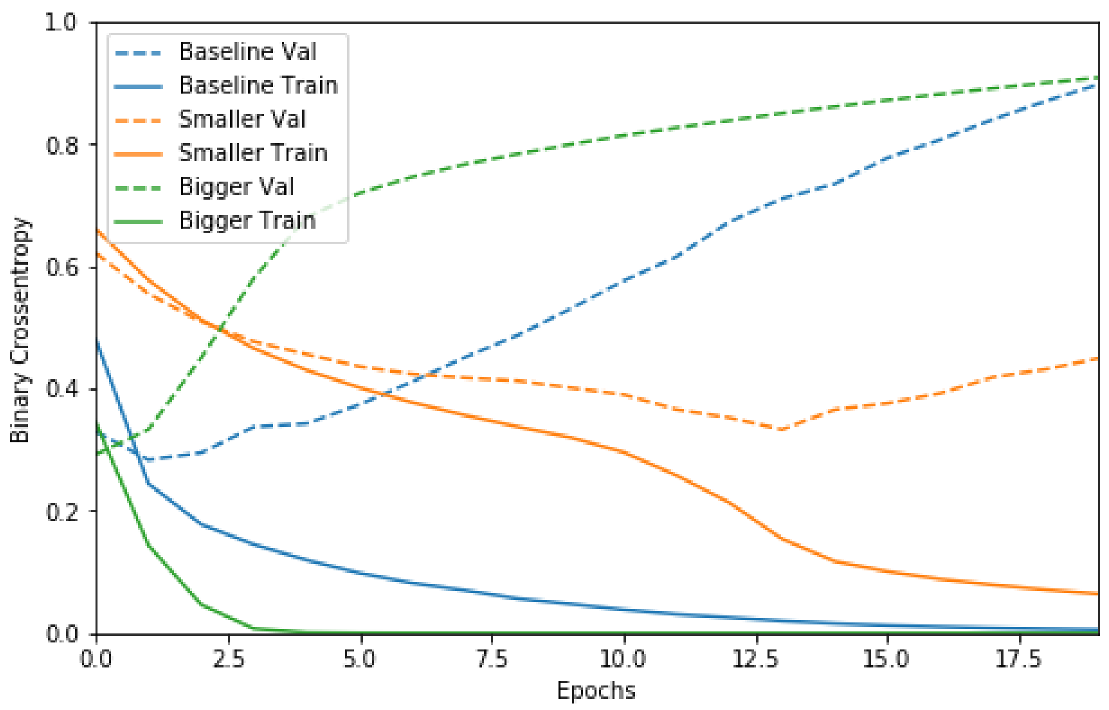
[9]:
```
bigger_history = bigger_model.fit(train_data, train_labels,
                                  epochs=20,
                                  batch_size=512,
                                  validation_data=(test_data, test_labels),
                                  verbose=2)
```

```
Train on 25000 samples, validate on 25000 samples
Epoch 1/20
 - 18s - loss: 0.3461 - acc: 0.8520 - binary_crossentropy: 0.3461 - val_loss:
0.2922 - val_acc: 0.8817 - val_binary_crossentropy: 0.2922
```

```
Epoch 2/20
 - 14s - loss: 0.1437 - acc: 0.9478 - binary_crossentropy: 0.1437 - val_loss:
0.3330 - val_acc: 0.8723 - val_binary_crossentropy: 0.3330
Epoch 3/20
 - 13s - loss: 0.0465 - acc: 0.9872 - binary_crossentropy: 0.0465 - val_loss:
0.4505 - val_acc: 0.8696 - val_binary_crossentropy: 0.4505
Epoch 4/20
 - 13s - loss: 0.0072 - acc: 0.9986 - binary_crossentropy: 0.0072 - val_loss:
0.5805 - val_acc: 0.8685 - val_binary_crossentropy: 0.5805
Epoch 5/20
 - 14s - loss: 9.1697e-04 - acc: 1.0000 - binary_crossentropy: 9.1697e-04 -
val_loss: 0.6804 - val_acc: 0.8703 - val_binary_crossentropy: 0.6804
Epoch 6/20
 - 16s - loss: 2.4898e-04 - acc: 1.0000 - binary_crossentropy: 2.4898e-04 -
val_loss: 0.7200 - val_acc: 0.8706 - val_binary_crossentropy: 0.7200
Epoch 7/20
 - 13s - loss: 1.5187e-04 - acc: 1.0000 - binary_crossentropy: 1.5187e-04 -
val_loss: 0.7462 - val_acc: 0.8705 - val_binary_crossentropy: 0.7462
Epoch 8/20
 - 14s - loss: 1.0835e-04 - acc: 1.0000 - binary_crossentropy: 1.0835e-04 -
val_loss: 0.7670 - val_acc: 0.8704 - val_binary_crossentropy: 0.7670
Epoch 9/20
 - 13s - loss: 8.1950e-05 - acc: 1.0000 - binary_crossentropy: 8.1950e-05 -
val_loss: 0.7837 - val_acc: 0.8706 - val_binary_crossentropy: 0.7837
Epoch 10/20
 - 13s - loss: 6.3951e-05 - acc: 1.0000 - binary_crossentropy: 6.3951e-05 -
val_loss: 0.7994 - val_acc: 0.8708 - val_binary_crossentropy: 0.7994
Epoch 11/20
 - 14s - loss: 5.0951e-05 - acc: 1.0000 - binary_crossentropy: 5.0951e-05 -
val_loss: 0.8140 - val_acc: 0.8708 - val_binary_crossentropy: 0.8140
Epoch 12/20
 - 13s - loss: 4.1561e-05 - acc: 1.0000 - binary_crossentropy: 4.1561e-05 -
val_loss: 0.8266 - val_acc: 0.8707 - val_binary_crossentropy: 0.8266
Epoch 13/20
 - 13s - loss: 3.4086e-05 - acc: 1.0000 - binary_crossentropy: 3.4086e-05 -
val_loss: 0.8384 - val_acc: 0.8709 - val_binary_crossentropy: 0.8384
Epoch 14/20
 - 13s - loss: 2.8378e-05 - acc: 1.0000 - binary_crossentropy: 2.8378e-05 -
val_loss: 0.8502 - val_acc: 0.8708 - val_binary_crossentropy: 0.8502
Epoch 15/20
 - 13s - loss: 2.3815e-05 - acc: 1.0000 - binary_crossentropy: 2.3815e-05 -
val_loss: 0.8611 - val_acc: 0.8708 - val_binary_crossentropy: 0.8611
Epoch 16/20
 - 14s - loss: 2.0211e-05 - acc: 1.0000 - binary_crossentropy: 2.0211e-05 -
val_loss: 0.8716 - val_acc: 0.8708 - val_binary_crossentropy: 0.8716
Epoch 17/20
 - 13s - loss: 1.7215e-05 - acc: 1.0000 - binary_crossentropy: 1.7215e-05 -
val_loss: 0.8816 - val_acc: 0.8708 - val_binary_crossentropy: 0.8816
```

```
Epoch 18/20
 - 13s - loss: 1.4778e-05 - acc: 1.0000 - binary_crossentropy: 1.4778e-05 -
val_loss: 0.8910 - val_acc: 0.8708 - val_binary_crossentropy: 0.8910
Epoch 19/20
 - 14s - loss: 1.2767e-05 - acc: 1.0000 - binary_crossentropy: 1.2767e-05 -
val_loss: 0.9000 - val_acc: 0.8706 - val_binary_crossentropy: 0.9000
Epoch 20/20
 - 13s - loss: 1.1097e-05 - acc: 1.0000 - binary_crossentropy: 1.1097e-05 -
val_loss: 0.9085 - val_acc: 0.8706 - val_binary_crossentropy: 0.9085
```

[17]:
```
def plot_history(histories, key='binary_crossentropy', y_min=0):
    plt.figure(figsize=(8,5))

    for name, history in histories:
        val = plt.plot(history.epoch, history.history['val_'+key],
                       '--', label=name.title()+' Val')
        plt.plot(history.epoch, history.history[key], color=val[0].get_color(),
                 label=name.title()+' Train')

    plt.xlabel('Epochs')
    plt.ylabel(key.replace('_',' ').title())
    plt.legend()

    plt.xlim([0,max(history.epoch)])
    plt.ylim([y_min, 1])


plot_history([('baseline', baseline_history),
              ('smaller', smaller_history),
              ('bigger', bigger_history)])
```



The bigger network starts to overfit after one epoch and much more severly. The smaller starts to overfit after the baseline and more slowly. So more capacity, means low training loss and high suceptibility to overfit (large difference between training and validation loss).

1.3 Strategies

1.3.1 Weight regularization

The simpler models are usually the best against overfitting. So if it has less parameters or if they have less entropy it is simpler. Thus forcing the weights to take small values, makes the distribution more "regular".

This is done by adding to the loss function a cost associated to large weights. There are two possibilities: - L1 regularization: cost added is proportional to the weights (L1 norms) - L2 regularization (weight decay): cost added is proportional to the square of the weights (L2 norms)

[11]:
```
l2_model = keras.models.Sequential([
    keras.layers.Dense(16, kernel_regularizer=keras.regularizers.l2(0.001),
                       activation=tf.nn.relu, input_shape=(NUM_WORDS,)),
    keras.layers.Dense(16, kernel_regularizer=keras.regularizers.l2(0.001),
                       activation=tf.nn.relu),
    keras.layers.Dense(1, activation=tf.nn.sigmoid)
])

l2_model.compile(optimizer='adam',
                 loss='binary_crossentropy',
                 metrics=['accuracy', 'binary_crossentropy'])
```

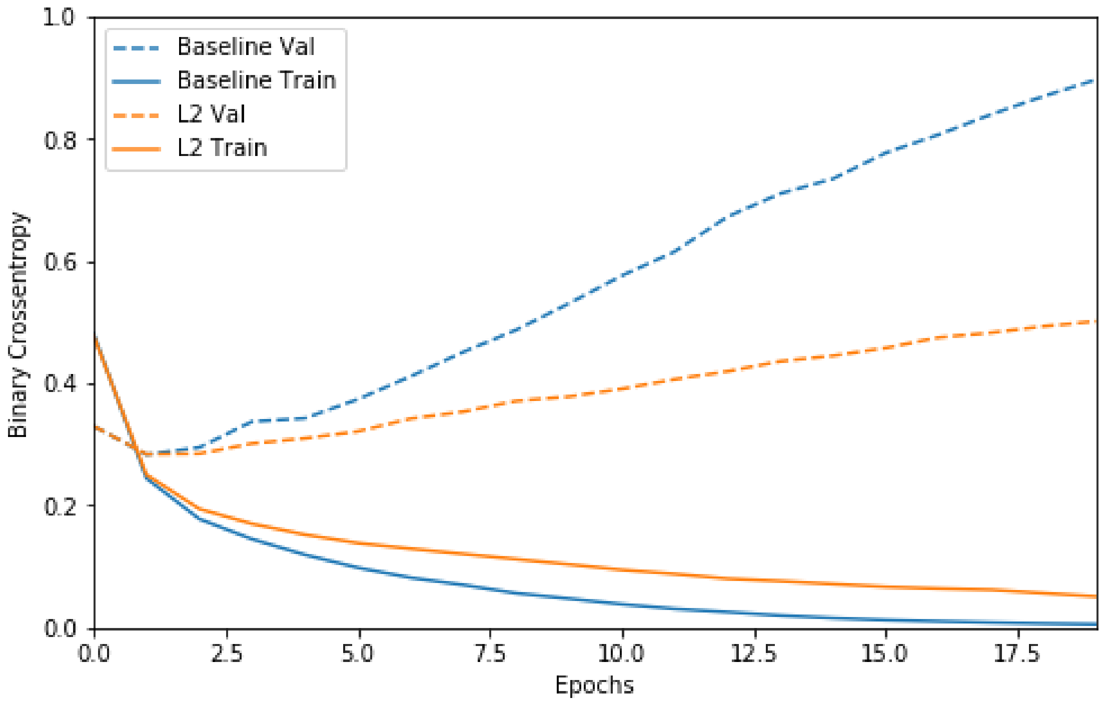
```
l2_model_history = l2_model.fit(train_data, train_labels,
                                epochs=20,
                                batch_size=512,
                                validation_data=(test_data, test_labels),
                                verbose=2)
```

```
Train on 25000 samples, validate on 25000 samples
Epoch 1/20
 - 6s - loss: 0.5198 - acc: 0.8112 - binary_crossentropy: 0.4784 - val_loss:
0.3732 - val_acc: 0.8774 - val_binary_crossentropy: 0.3301
Epoch 2/20
 - 5s - loss: 0.2977 - acc: 0.9102 - binary_crossentropy: 0.2501 - val_loss:
0.3349 - val_acc: 0.8878 - val_binary_crossentropy: 0.2841
Epoch 3/20
 - 5s - loss: 0.2475 - acc: 0.9320 - binary_crossentropy: 0.1942 - val_loss:
0.3401 - val_acc: 0.8849 - val_binary_crossentropy: 0.2850
Epoch 4/20
 - 5s - loss: 0.2269 - acc: 0.9418 - binary_crossentropy: 0.1699 - val_loss:
0.3599 - val_acc: 0.8784 - val_binary_crossentropy: 0.3015
Epoch 5/20
 - 5s - loss: 0.2117 - acc: 0.9492 - binary_crossentropy: 0.1522 - val_loss:
0.3705 - val_acc: 0.8762 - val_binary_crossentropy: 0.3100
Epoch 6/20
 - 5s - loss: 0.2000 - acc: 0.9544 - binary_crossentropy: 0.1384 - val_loss:
0.3836 - val_acc: 0.8749 - val_binary_crossentropy: 0.3213
Epoch 7/20
 - 5s - loss: 0.1924 - acc: 0.9577 - binary_crossentropy: 0.1291 - val_loss:
0.4065 - val_acc: 0.8711 - val_binary_crossentropy: 0.3423
Epoch 8/20
 - 5s - loss: 0.1856 - acc: 0.9596 - binary_crossentropy: 0.1205 - val_loss:
0.4195 - val_acc: 0.8698 - val_binary_crossentropy: 0.3536
Epoch 9/20
 - 5s - loss: 0.1790 - acc: 0.9624 - binary_crossentropy: 0.1120 - val_loss:
0.4389 - val_acc: 0.8647 - val_binary_crossentropy: 0.3711
Epoch 10/20
 - 5s - loss: 0.1719 - acc: 0.9665 - binary_crossentropy: 0.1035 - val_loss:
0.4474 - val_acc: 0.8663 - val_binary_crossentropy: 0.3783
Epoch 11/20
 - 5s - loss: 0.1641 - acc: 0.9704 - binary_crossentropy: 0.0946 - val_loss:
0.4612 - val_acc: 0.8653 - val_binary_crossentropy: 0.3910
Epoch 12/20
 - 5s - loss: 0.1585 - acc: 0.9734 - binary_crossentropy: 0.0878 - val_loss:
0.4777 - val_acc: 0.8626 - val_binary_crossentropy: 0.4066
Epoch 13/20
 - 5s - loss: 0.1517 - acc: 0.9768 - binary_crossentropy: 0.0802 - val_loss:
0.4913 - val_acc: 0.8614 - val_binary_crossentropy: 0.4195
```

```
Epoch 14/20
 - 5s - loss: 0.1486 - acc: 0.9773 - binary_crossentropy: 0.0762 - val_loss:
0.5088 - val_acc: 0.8591 - val_binary_crossentropy: 0.4360
Epoch 15/20
 - 5s - loss: 0.1444 - acc: 0.9799 - binary_crossentropy: 0.0714 - val_loss:
0.5186 - val_acc: 0.8588 - val_binary_crossentropy: 0.4452
Epoch 16/20
 - 5s - loss: 0.1409 - acc: 0.9808 - binary_crossentropy: 0.0668 - val_loss:
0.5322 - val_acc: 0.8578 - val_binary_crossentropy: 0.4579
Epoch 17/20
 - 5s - loss: 0.1382 - acc: 0.9819 - binary_crossentropy: 0.0641 - val_loss:
0.5494 - val_acc: 0.8566 - val_binary_crossentropy: 0.4749
Epoch 18/20
 - 5s - loss: 0.1371 - acc: 0.9826 - binary_crossentropy: 0.0619 - val_loss:
0.5586 - val_acc: 0.8564 - val_binary_crossentropy: 0.4829
Epoch 19/20
 - 5s - loss: 0.1317 - acc: 0.9842 - binary_crossentropy: 0.0563 - val_loss:
0.5690 - val_acc: 0.8562 - val_binary_crossentropy: 0.4934
Epoch 20/20
 - 5s - loss: 0.1263 - acc: 0.9876 - binary_crossentropy: 0.0509 - val_loss:
0.5766 - val_acc: 0.8548 - val_binary_crossentropy: 0.5014
```

l2(0.001) means that every weight in the weight matrix will add 0.001wˆ2 to the total loss. Because of this the loss during training is much higher than during testing.

[18]:
```
plot_history([('baseline', baseline_history),
              ('l2', l2_model_history)])
```



1.3.2 Dropout

One of the most effective regularization techniques. It consistis on dropping out (i.e. set to zero) some output features of the layer during training randomly.

Ex.: A layer output [0.2, 0.5, 1.3, 0.8, 1.1] => [0, 0.5, 1.3, 0, 1.1] (random case)

The "dropout rate" is the fraction of features to be set to zero (usually between [0.2, 0.5]). This is not done during testing, instead outputs are scaled down by the dropout rate to compensate.

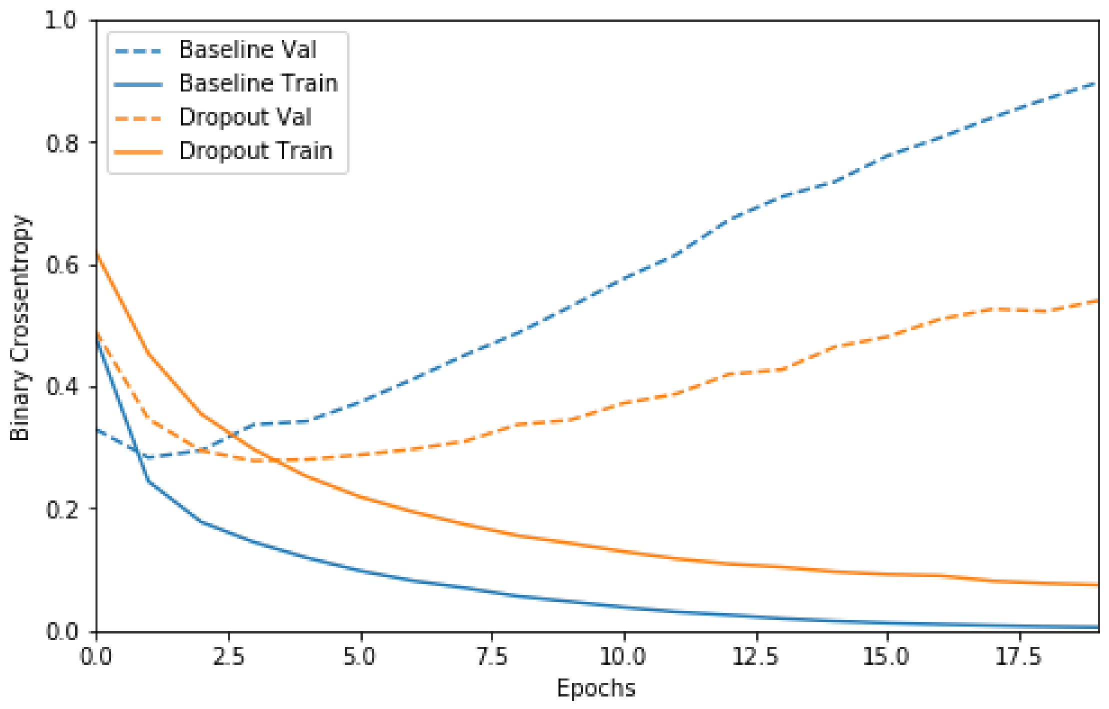
[13]:
```
dpt_model = keras.models.Sequential([
    keras.layers.Dense(16, activation=tf.nn.relu, input_shape=(NUM_WORDS,)),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(16, activation=tf.nn.relu),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(1, activation=tf.nn.sigmoid)
])

dpt_model.compile(optimizer='adam',
                  loss='binary_crossentropy',
                  metrics=['accuracy','binary_crossentropy'])

dpt_model_history = dpt_model.fit(train_data, train_labels,
                                  epochs=20,
                                  batch_size=512,
                                  validation_data=(test_data, test_labels),
                                  verbose=2)
```

```
WARNING:tensorflow:From C:\Users\Pedro\AppData\Roaming\Python\Python37\site-
packages\tensorflow\python\keras\layers\core.py:143: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed
in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 -
keep_prob`.
Train on 25000 samples, validate on 25000 samples
Epoch 1/20
 - 6s - loss: 0.6203 - acc: 0.6558 - binary_crossentropy: 0.6203 - val_loss:
0.4907 - val_acc: 0.8448 - val_binary_crossentropy: 0.4907
Epoch 2/20
 - 7s - loss: 0.4535 - acc: 0.8092 - binary_crossentropy: 0.4535 - val_loss:
0.3468 - val_acc: 0.8820 - val_binary_crossentropy: 0.3468
Epoch 3/20
 - 6s - loss: 0.3546 - acc: 0.8693 - binary_crossentropy: 0.3546 - val_loss:
0.2948 - val_acc: 0.8883 - val_binary_crossentropy: 0.2948
Epoch 4/20
 - 6s - loss: 0.2969 - acc: 0.9010 - binary_crossentropy: 0.2969 - val_loss:
```

```
0.2782 - val_acc: 0.8892 - val_binary_crossentropy: 0.2782
Epoch 5/20
 - 7s - loss: 0.2524 - acc: 0.9196 - binary_crossentropy: 0.2524 - val_loss:
0.2805 - val_acc: 0.8892 - val_binary_crossentropy: 0.2805
Epoch 6/20
 - 8s - loss: 0.2191 - acc: 0.9317 - binary_crossentropy: 0.2191 - val_loss:
0.2879 - val_acc: 0.8834 - val_binary_crossentropy: 0.2879
Epoch 7/20
 - 6s - loss: 0.1945 - acc: 0.9401 - binary_crossentropy: 0.1945 - val_loss:
0.2971 - val_acc: 0.8840 - val_binary_crossentropy: 0.2971
Epoch 8/20
 - 6s - loss: 0.1738 - acc: 0.9464 - binary_crossentropy: 0.1738 - val_loss:
0.3103 - val_acc: 0.8824 - val_binary_crossentropy: 0.3103
Epoch 9/20
 - 6s - loss: 0.1554 - acc: 0.9532 - binary_crossentropy: 0.1554 - val_loss:
0.3375 - val_acc: 0.8818 - val_binary_crossentropy: 0.3375
Epoch 10/20
 - 6s - loss: 0.1429 - acc: 0.9562 - binary_crossentropy: 0.1429 - val_loss:
0.3451 - val_acc: 0.8776 - val_binary_crossentropy: 0.3451
Epoch 11/20
 - 6s - loss: 0.1294 - acc: 0.9615 - binary_crossentropy: 0.1294 - val_loss:
0.3725 - val_acc: 0.8785 - val_binary_crossentropy: 0.3725
Epoch 12/20
 - 6s - loss: 0.1177 - acc: 0.9644 - binary_crossentropy: 0.1177 - val_loss:
0.3877 - val_acc: 0.8769 - val_binary_crossentropy: 0.3877
Epoch 13/20
 - 6s - loss: 0.1092 - acc: 0.9668 - binary_crossentropy: 0.1092 - val_loss:
0.4198 - val_acc: 0.8743 - val_binary_crossentropy: 0.4198
Epoch 14/20
 - 6s - loss: 0.1040 - acc: 0.9680 - binary_crossentropy: 0.1040 - val_loss:
0.4274 - val_acc: 0.8741 - val_binary_crossentropy: 0.4274
Epoch 15/20
 - 7s - loss: 0.0964 - acc: 0.9712 - binary_crossentropy: 0.0964 - val_loss:
0.4645 - val_acc: 0.8758 - val_binary_crossentropy: 0.4645
Epoch 16/20
 - 7s - loss: 0.0921 - acc: 0.9723 - binary_crossentropy: 0.0921 - val_loss:
0.4812 - val_acc: 0.8754 - val_binary_crossentropy: 0.4812
Epoch 17/20
 - 6s - loss: 0.0899 - acc: 0.9740 - binary_crossentropy: 0.0899 - val_loss:
0.5100 - val_acc: 0.8745 - val_binary_crossentropy: 0.5100
Epoch 18/20
 - 6s - loss: 0.0813 - acc: 0.9767 - binary_crossentropy: 0.0813 - val_loss:
0.5266 - val_acc: 0.8751 - val_binary_crossentropy: 0.5266
Epoch 19/20
 - 6s - loss: 0.0773 - acc: 0.9782 - binary_crossentropy: 0.0773 - val_loss:
0.5228 - val_acc: 0.8737 - val_binary_crossentropy: 0.5228
Epoch 20/20
 - 6s - loss: 0.0748 - acc: 0.9785 - binary_crossentropy: 0.0748 - val_loss:
```

```
0.5404 - val_acc: 0.8720 - val_binary_crossentropy: 0.5404
```

[19]:
```python
plot_history([('baseline', baseline_history),
              ('dropout', dpt_model_history)])
```



1.3.3 Combining both strategies

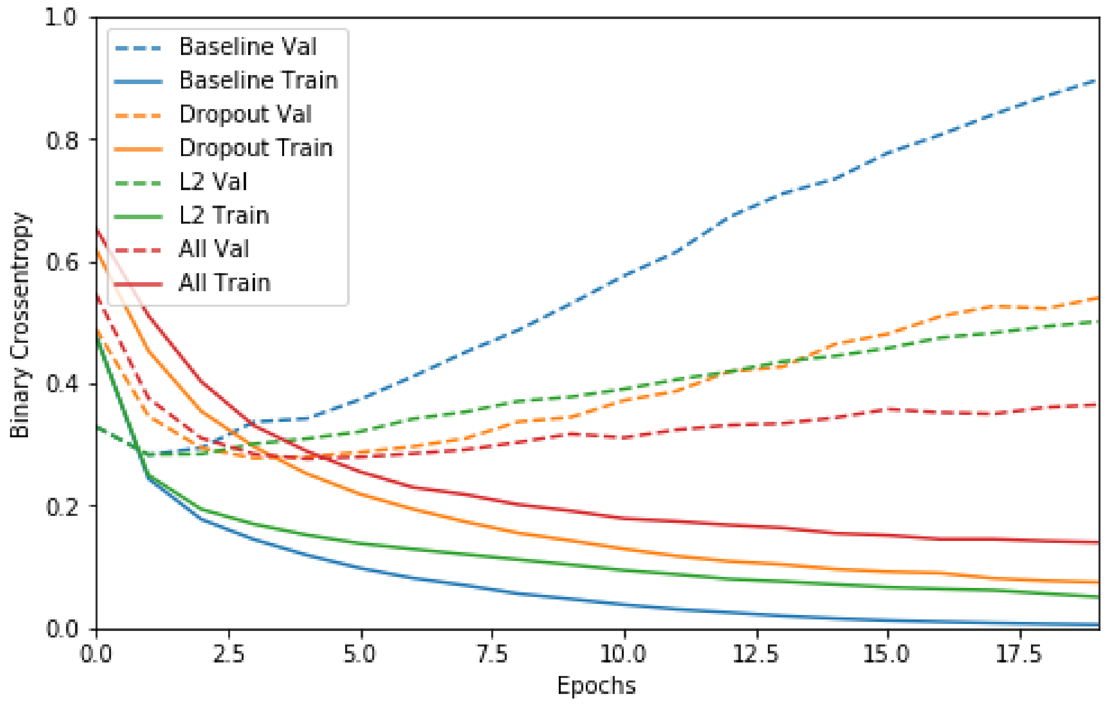
[15]:
```python
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=3)

all_model = keras.models.Sequential([
    keras.layers.Dense(16, kernel_regularizer=keras.regularizers.l2(0.001),␣
 ↪activation=tf.nn.relu, input_shape=(NUM_WORDS,)),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(16, kernel_regularizer=keras.regularizers.l2(0.001),␣
 ↪activation=tf.nn.relu),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(1, activation=tf.nn.sigmoid)
])

all_model.compile(optimizer='adam',
                  loss='binary_crossentropy',
                  metrics=['accuracy','binary_crossentropy'])
```

```python
all_model_history = all_model.fit(train_data, train_labels,
                                  epochs=20,
                                  batch_size=512,
                                  validation_data=(test_data, test_labels),
                                  verbose=1)
                                  #callbacks=[early_stop])
```

```
Train on 25000 samples, validate on 25000 samples
Epoch 1/20
25000/25000 [==============================] - 7s 282us/sample - loss: 0.6899 -
acc: 0.5950 - binary_crossentropy: 0.6545 - val_loss: 0.5791 - val_acc: 0.8408 -
val_binary_crossentropy: 0.5476
Epoch 2/20
25000/25000 [==============================] - 7s 286us/sample - loss: 0.5444 -
acc: 0.7659 - binary_crossentropy: 0.5110 - val_loss: 0.4123 - val_acc: 0.8755 -
val_binary_crossentropy: 0.3762
Epoch 3/20
25000/25000 [==============================] - 7s 264us/sample - loss: 0.4426 -
acc: 0.8405 - binary_crossentropy: 0.4031 - val_loss: 0.3540 - val_acc: 0.8833 -
val_binary_crossentropy: 0.3109
Epoch 4/20
25000/25000 [==============================] - 7s 269us/sample - loss: 0.3776 -
acc: 0.8819 - binary_crossentropy: 0.3312 - val_loss: 0.3345 - val_acc: 0.8878 -
val_binary_crossentropy: 0.2850
Epoch 5/20
25000/25000 [==============================] - 6s 258us/sample - loss: 0.3419 -
acc: 0.9000 - binary_crossentropy: 0.2895 - val_loss: 0.3322 - val_acc: 0.8864 -
val_binary_crossentropy: 0.2770
Epoch 6/20
25000/25000 [==============================] - 6s 254us/sample - loss: 0.3133 -
acc: 0.9144 - binary_crossentropy: 0.2559 - val_loss: 0.3395 - val_acc: 0.8859 -
val_binary_crossentropy: 0.2800
Epoch 7/20
25000/25000 [==============================] - 7s 271us/sample - loss: 0.2916 -
acc: 0.9230 - binary_crossentropy: 0.2301 - val_loss: 0.3489 - val_acc: 0.8840 -
val_binary_crossentropy: 0.2856
Epoch 8/20
25000/25000 [==============================] - 7s 261us/sample - loss: 0.2832 -
acc: 0.9283 - binary_crossentropy: 0.2183 - val_loss: 0.3585 - val_acc: 0.8798 -
val_binary_crossentropy: 0.2920
Epoch 9/20
25000/25000 [==============================] - 6s 255us/sample - loss: 0.2699 -
acc: 0.9321 - binary_crossentropy: 0.2019 - val_loss: 0.3733 - val_acc: 0.8805 -
val_binary_crossentropy: 0.3040
Epoch 10/20
25000/25000 [==============================] - 6s 247us/sample - loss: 0.2620 -
acc: 0.9345 - binary_crossentropy: 0.1915 - val_loss: 0.3895 - val_acc: 0.8796 -
```

```
val_binary_crossentropy: 0.3178
Epoch 11/20
25000/25000 [==============================] - 6s 246us/sample - loss: 0.2519 -
acc: 0.9397 - binary_crossentropy: 0.1790 - val_loss: 0.3855 - val_acc: 0.8785 -
val_binary_crossentropy: 0.3116
Epoch 12/20
25000/25000 [==============================] - 6s 257us/sample - loss: 0.2491 -
acc: 0.9415 - binary_crossentropy: 0.1744 - val_loss: 0.3999 - val_acc: 0.8776 -
val_binary_crossentropy: 0.3245
Epoch 13/20
25000/25000 [==============================] - 6s 245us/sample - loss: 0.2445 -
acc: 0.9420 - binary_crossentropy: 0.1684 - val_loss: 0.4088 - val_acc: 0.8786 -
val_binary_crossentropy: 0.3321
Epoch 14/20
25000/25000 [==============================] - 6s 242us/sample - loss: 0.2412 -
acc: 0.9446 - binary_crossentropy: 0.1637 - val_loss: 0.4132 - val_acc: 0.8784 -
val_binary_crossentropy: 0.3349
Epoch 15/20
25000/25000 [==============================] - 6s 254us/sample - loss: 0.2342 -
acc: 0.9472 - binary_crossentropy: 0.1549 - val_loss: 0.4244 - val_acc: 0.8782 -
val_binary_crossentropy: 0.3442
Epoch 16/20
25000/25000 [==============================] - 6s 260us/sample - loss: 0.2322 -
acc: 0.9492 - binary_crossentropy: 0.1516 - val_loss: 0.4393 - val_acc: 0.8782 -
val_binary_crossentropy: 0.3583
Epoch 17/20
25000/25000 [==============================] - 6s 258us/sample - loss: 0.2268 -
acc: 0.9493 - binary_crossentropy: 0.1454 - val_loss: 0.4342 - val_acc: 0.8745 -
val_binary_crossentropy: 0.3526
Epoch 18/20
25000/25000 [==============================] - 7s 263us/sample - loss: 0.2276 -
acc: 0.9475 - binary_crossentropy: 0.1454 - val_loss: 0.4331 - val_acc: 0.8767 -
val_binary_crossentropy: 0.3505
Epoch 19/20
25000/25000 [==============================] - 6s 254us/sample - loss: 0.2252 -
acc: 0.9512 - binary_crossentropy: 0.1422 - val_loss: 0.4447 - val_acc: 0.8745 -
val_binary_crossentropy: 0.3613
Epoch 20/20
25000/25000 [==============================] - 6s 258us/sample - loss: 0.2239 -
acc: 0.9508 - binary_crossentropy: 0.1402 - val_loss: 0.4495 - val_acc: 0.8749 -
val_binary_crossentropy: 0.3656
```

[21]:
```
plot_history([('baseline', baseline_history),
              ('dropout', dpt_model_history),
              ('l2', l2_model_history),
              ('all', all_model_history)])
```



1.4 Conclusion

Adding dropout is a clear improvement over the baseline model.

To recap: here the most common ways to prevent overfitting in neural networks:

- Get more training data.
- Reduce the capacity of the network.
- Add weight regularization.
- Add dropout.
- And two important approaches not covered in this notebook are data-augmentation and batch normalization.