Neural Network Regularization
Data Augmentation

CS109A Introduction to Data Science

Pavlos Protopapas, Kevin Rader and Chris Tanner

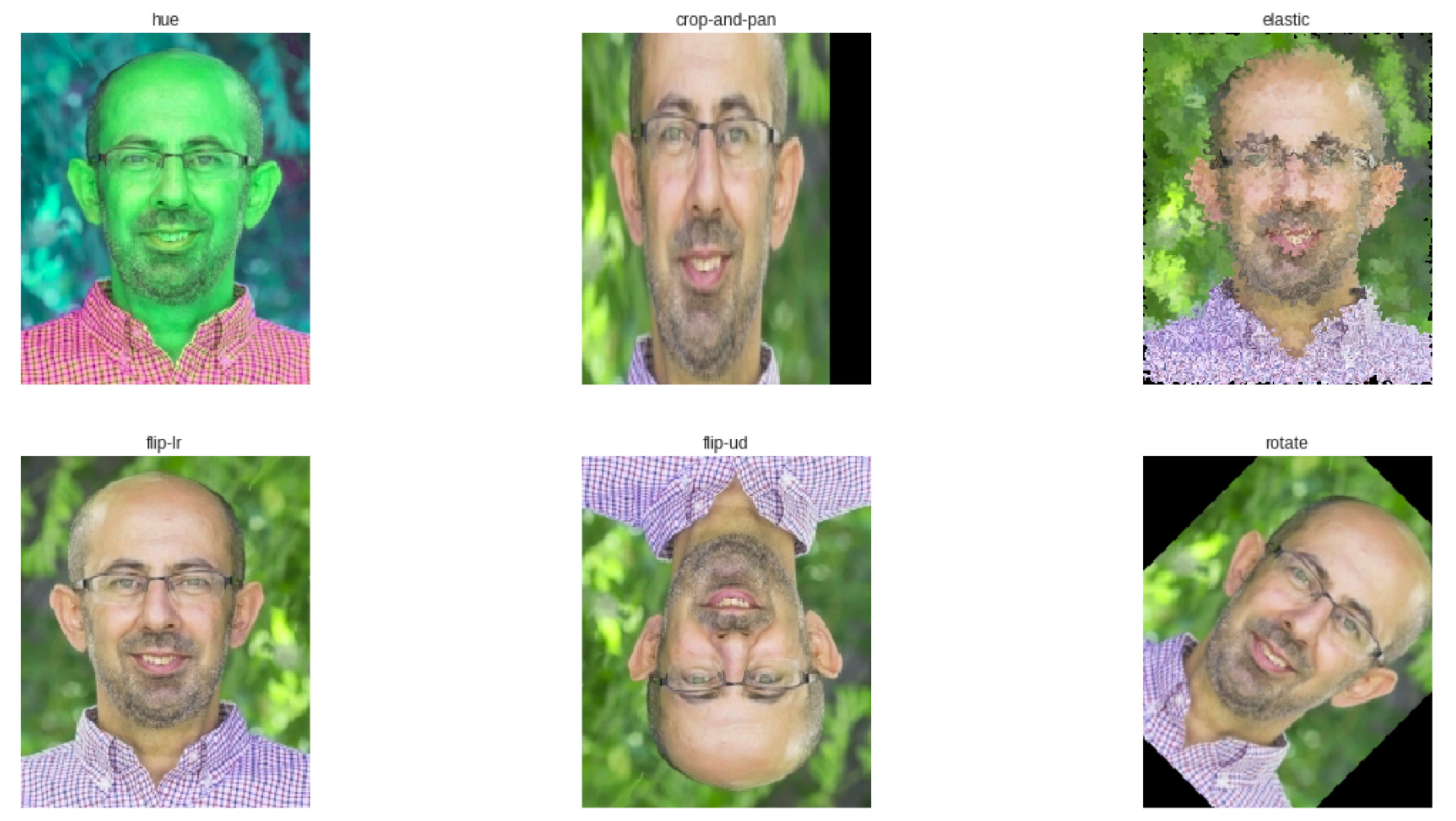
Outline

Regularization of NN

- Norm Penalties
- Early Stopping
- **Data Augmentation**
- Dropout


When you move
on to Deep Learning
Thanks guys
@datascienceinfo
Decision Tree
Linear Regression
K-Means
SVM
Random Forest
Naive Bayes
kNN
Logistic Regression
XGBoost
So long, partner

Data Augmentation



Data Augmentation: dos and don'ts

We use ImageDataGenerator to augment the dataset

```python
def get_generator():
    # create duplicate images
    BATCHES_PER_EPOCH = 300//BATCH_SIZE
    classes = ['pavlos', 'not-pavlos']
    for img_class in classes:
        img = Image.open((f'{DATA_DIR}/{img_class}.jpeg'))
        for i in range(1, BATCH_SIZE*BATCHES_PER_EPOCH//2+1):
            img.thumbnail(TARGET_SIZE, Image.ANTIALIAS)
            img.save(f'{DATA_DIR}/{img_class}/{img_class}{i:0>3}.jpeg', "JPEG")

    data_gen = ImageDataGenerator(
            rescale=1./255,
            height_shift_range=0.5,
            width_shift_range=0.5)

    img_generator = data_gen.flow_from_directory(
            DATA_DIR,
            target_size=(TARGET_SIZE),
            batch_size=BATCH_SIZE,
            classes=classes,
            class_mode='binary')
    return img_generator
```

Data Augmentation: dos and don'ts

Carefully choose your transformations. Not all transformations are valid.

Label 6

Rotate by 180

Label 6

Data Augmentation does not work for tabular data and not as nicely for time series.

Outline

Regularization of NN

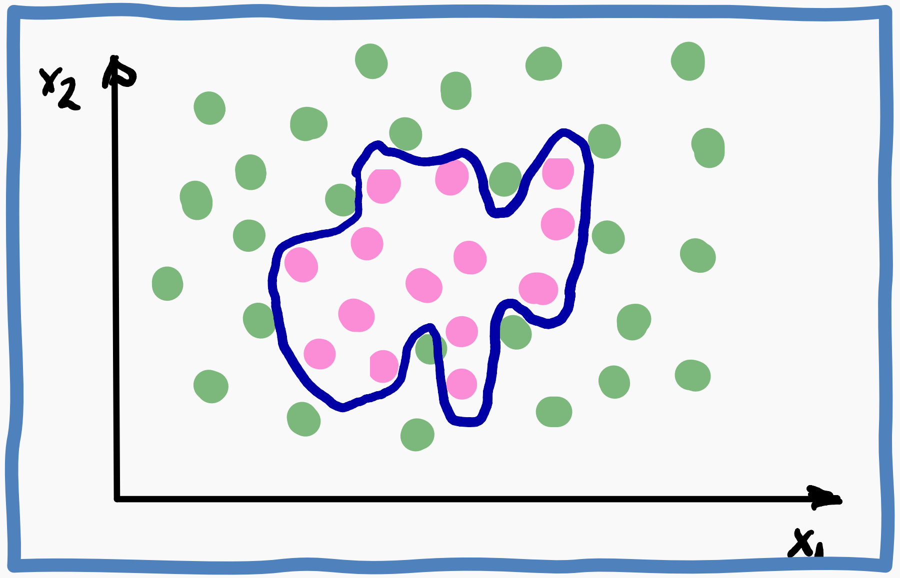
- Norm Penalties
- Early Stopping
- Data Augmentation
- **Dropout**

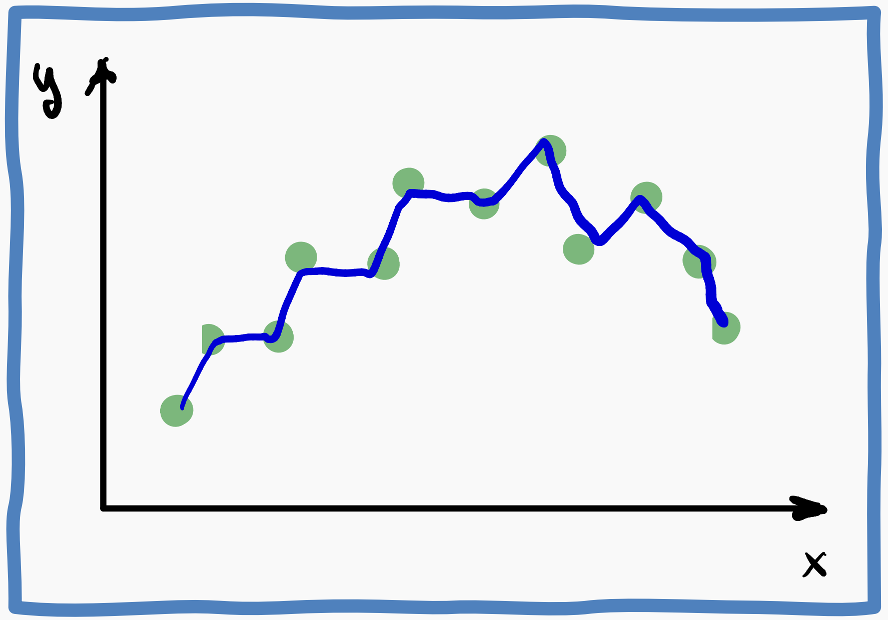
Co-adaptation

Overfitting occurs when the model is sensitive to slight variations on the input and therefore it fits the noise.

L1 and L2 regularizations 'shrink' the weights to avoid this problem.

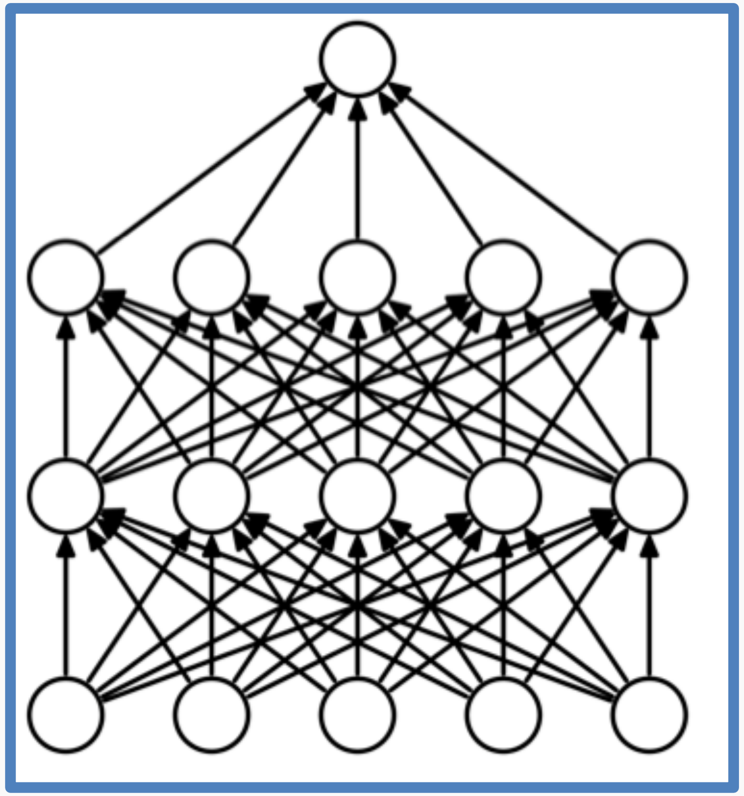
However in a large network many units can collaborate to respond to the input while the weights can remain relatively small. This is called co-adaptation.



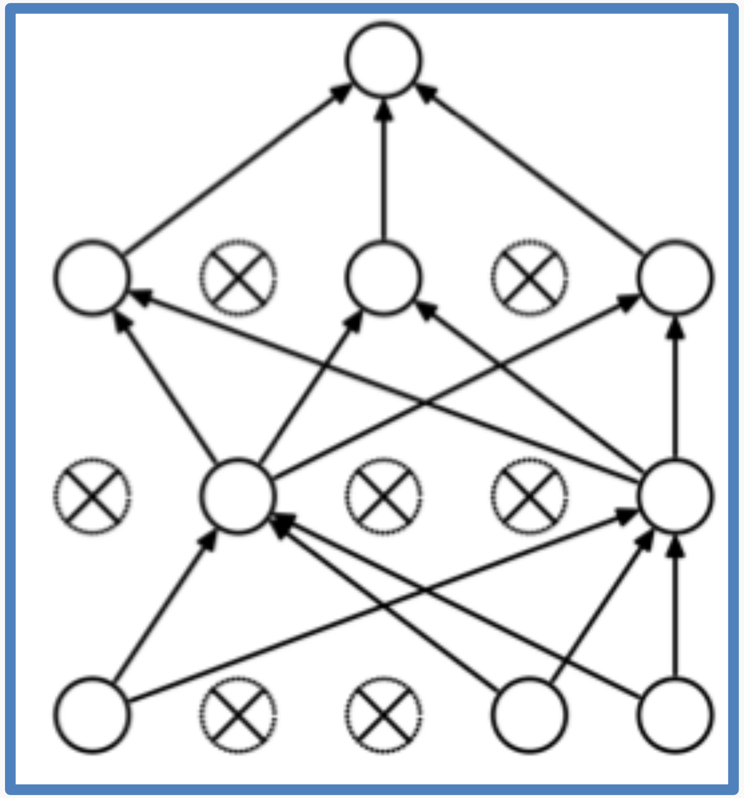


Dropout

- Randomly set some neurons and their connections to zero (i.e. “dropped”)
- Prevent overfitting by reducing co-adaptation of neurons
- Like training many random sub-networks



Standard Neural Network



After applying dropout

https://arxiv.org/pdf/1207.0580.pdf

Dropout: Training

For each new example in a mini-batch (could be for one mini-batch depending on the implementation):

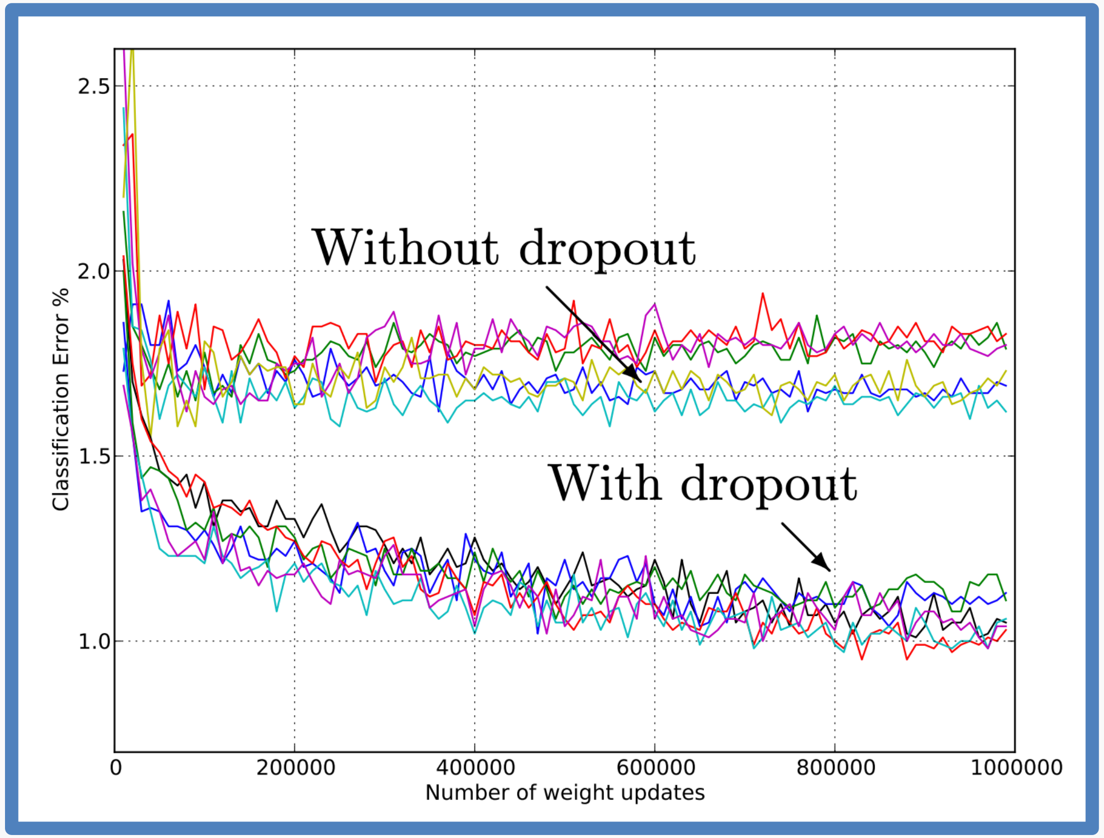
- Randomly sample a binary mask μ independently, where μ_i indicates if input/hidden node i is included
- Multiply output of node i with μ_i, and perform gradient update

Typically:

- Input nodes are <u>included</u> with prob=0.8 (as per original paper, but rarely used)
- Hidden nodes are <u>included</u> with prob=0.5

Dropout

- Widely used and highly effective



Test error for different architectures with and without dropout.

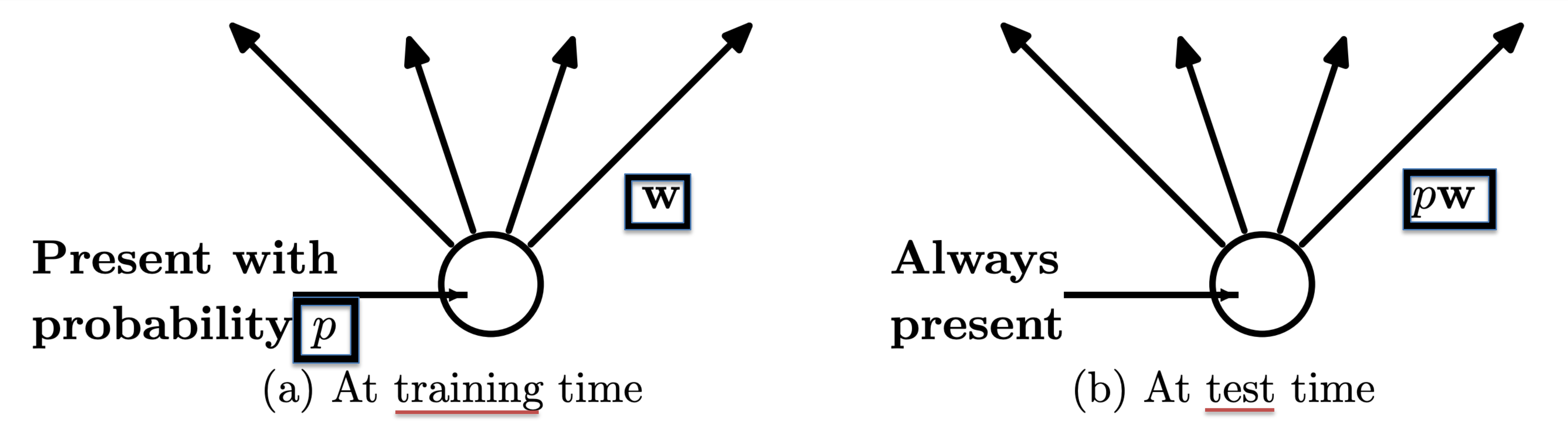
The networks have 2 to 4 hidden layers each with 1024 to 2048 units.

- Proposed as an alternative to ensemble methods, which is too expensive for neural nets

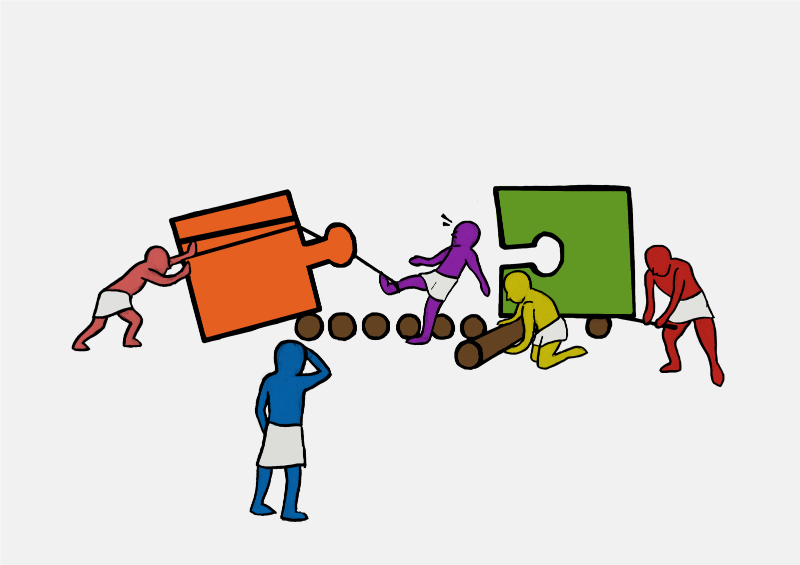
http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Dropout: Prediction

- We can think of dropout as training many of sub-networks
- At **test time**, we can “aggregate” over these sub-networks by **reducing connection weights in proportion to dropout probability, *p***



NOTE: Dropouts can be used for **neural network inference** by dropping during predictions and predicting multiple times to get a distribution



Exercise: Dropout