Simple Domain Adaptation in Neural Networks

Lecture#9

Domain Adaptation

- Training data comes in various styles, genre and domains
  - News, tweets, TED talks, Parliament proceedings
- A system built on one domain may not perform well on other domains
- Several times, annotated data of the domain that we care about (in-domain) is small but there is large heterogeneous out-of-domain data

Domain Adaptation

Domain adaptation aims to use all the available data for the benefit of the in-domain data

Domain Adaptation

Let’s consider a scenario

- When a crisis happens in the world, several NGO’s analyze Twitter feeds to identify needs in the crisis area such as, infrastructure damage, food scarcity
- The NGO’s wait for volunteers to annotate tweets, then build a classifier and automatically classify tweets into classes

Domain Adaptation

- Annotation is a time expensive process in terms of both time and money
- Time is critical especially in the beginning of the crisis

Domain Adaptation

- Annotation is a time expensive process in terms of both time and money
- Time is critical especially in the beginning of the crisis

- Can we make use of the data from previous disaster to speed up the whole process?

Domain Adaptation

Fine tuning / Extended training

- Neural models train in batches
- Once model matures, we can extend training with a different data
- This results in fine-tuning the model parameters towards the new training data
- Essentially, we are resuming the training of an already trained model for a few more epochs but on a different data

Domain Adaptation

- We can train an initial classifier on the previously available data (out-of-domain)
- At the time of a new disaster, this already trained classifier can be used to classify initial tweets
- Once annotated data arrives, we can simply extend the training of our current model on the new data
- Our model parameters will adjust according to the new data

Domain Adaptation

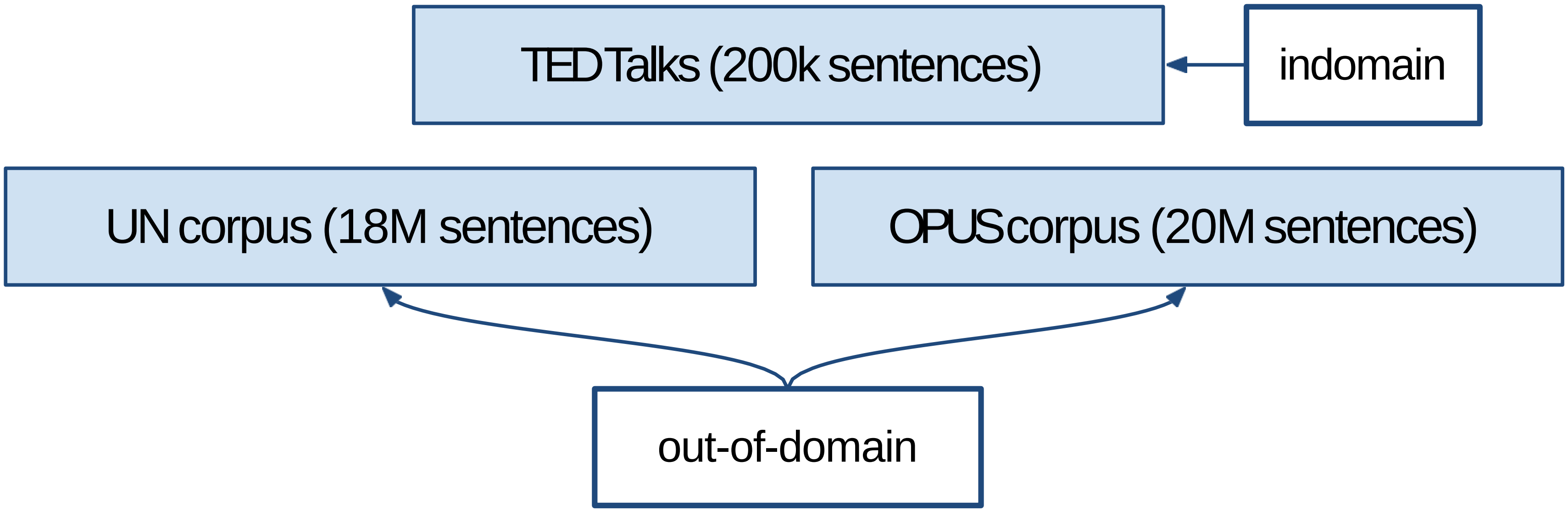
- The performance of the classifier when fine-tuned on the in-domain data improves significantly
- Fine-tuning a model is super efficient compared to training a model from scratch
- The resulting model is robust since it has seen a large amount of out-of-domain data plus some in-domain data.

Domain Adaptation

- Let's take machine translation as an example

Domain Adaptation

- Suppose, we want to train a neural machine translation (NMT) model for TED talks and there are three corpora available for training



Domain Adaptation

- Suppose, we want to train a neural machine translation (NMT) model for TED talks and there are three corpora available for training

TED Talks (200k sentences)

UN corpus (18M sentences)

OPUS corpus (20M sentences)

- What would be a good training strategy in order to built a good domain specific system?

Domain Adaptation

- What would be a good training strategy in order to built a good TED talks system?

- Use only TED talks because that's what we care about!

TED Talks (200k sentences)

UN corpus (18M sentences)

OPUS corpus (20M sentences)

Domain Adaptation

- What would be a good training strategy in order to built a good TED talks system?
- Use only TED talks because that's what we care about!

TED data is small. It would result in overfitting

TED Talks (200k sentences)

UN corpus (18M sentences)

OPUS corpus (20M sentences)

Domain Adaptation

- What would be a good training strategy in order to built a good TED talks system?

- Combine all three corpora together!

TED Talks (200k sentences)

UN corpus (18M sentences)

OPUS corpus (20M sentences)

Domain Adaptation

- What would be a good training strategy in order to built a good TED talks system?

- Combine all three corpora together!

TED is comparatively small. It will be lost in the concatenation of three corpora

TED Talks (200k sentences)

UN corpus (18M sentences)

OPUS corpus (20M sentences)

Domain Adaptation

- What would be a good training strategy in order to built a good TED talks system?

- **Solution:** Train a generic model on all the available data
- Later, **fine-tune** it on the in-domain data

TED Talks (200k sentences)

UN corpus (18M sentences)

OPUS corpus (20M sentences)

Domain Adaptation - Fine-tuning

- Consider an NMT training scenario that takes an input corpus and trains a model

Domain Adaptation - Fine-tuning

- Consider an NMT training scenario that takes an input corpus and trains a model
- First we train the model on the out-of-domain data for say, 20 epochs

Domain Adaptation - Fine-tuning

- Consider an NMT training scenario that takes an input corpus and trains a model
- First we train the model on the out-of-domain data for say, 20 epochs
- Now model has learned to translate language from a generic data

Domain Adaptation - Fine-tuning

- Consider an NMT training scenario that takes an input corpus and trains a model
- First we train the model on the out-of-domain data for say, 20 epochs
- Now model has learned to translate language from a generic data
- Let's fine-tune the model to our domain of interest which is TED talks!

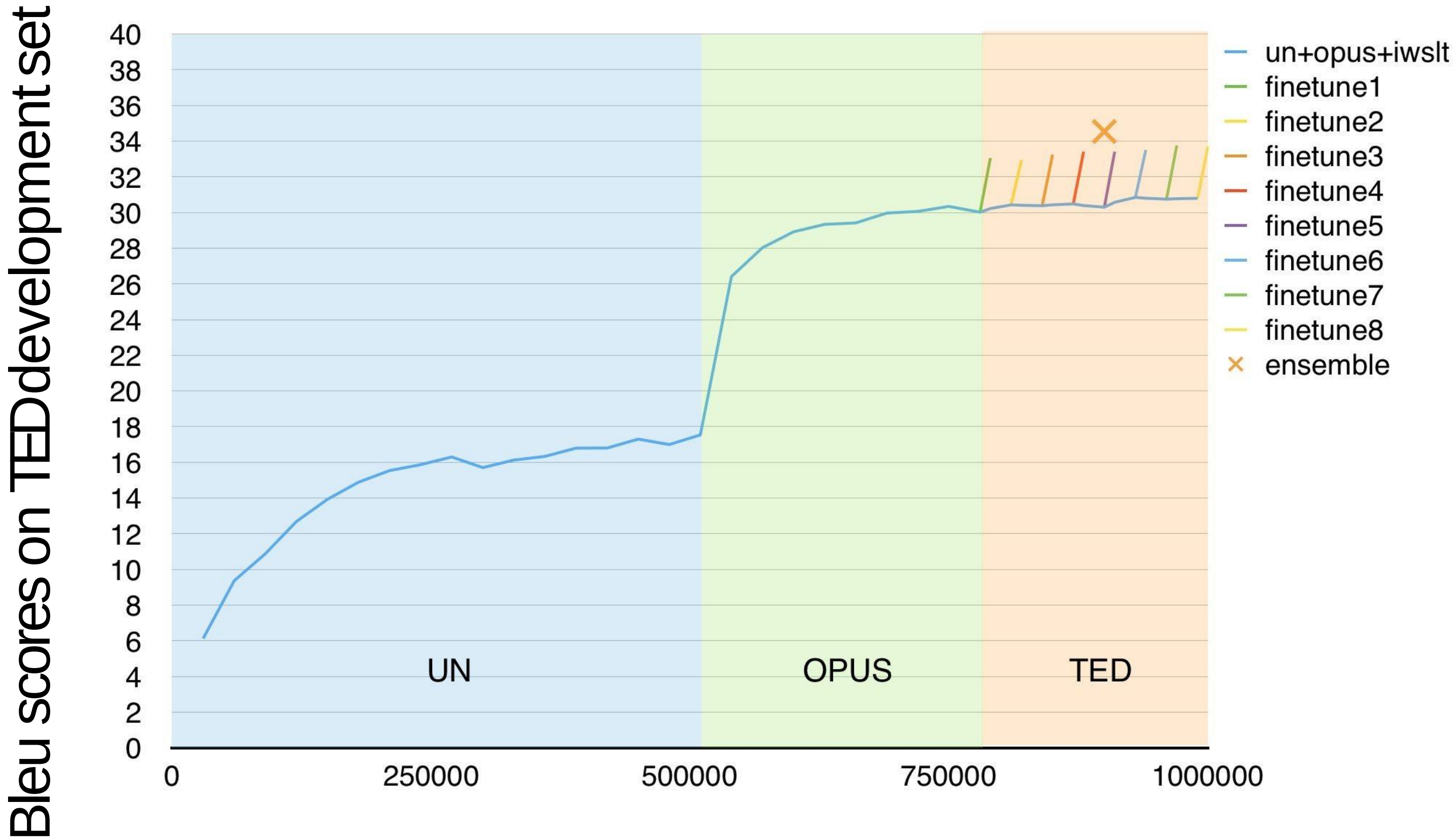
Domain Adaptation - Fine-tuning

- Remember, NMT runs in mini-batches. Essentially we are simply training an NMT system on a few more mini-batches but of TED corpus

Domain Adaptation - Fine-tuning

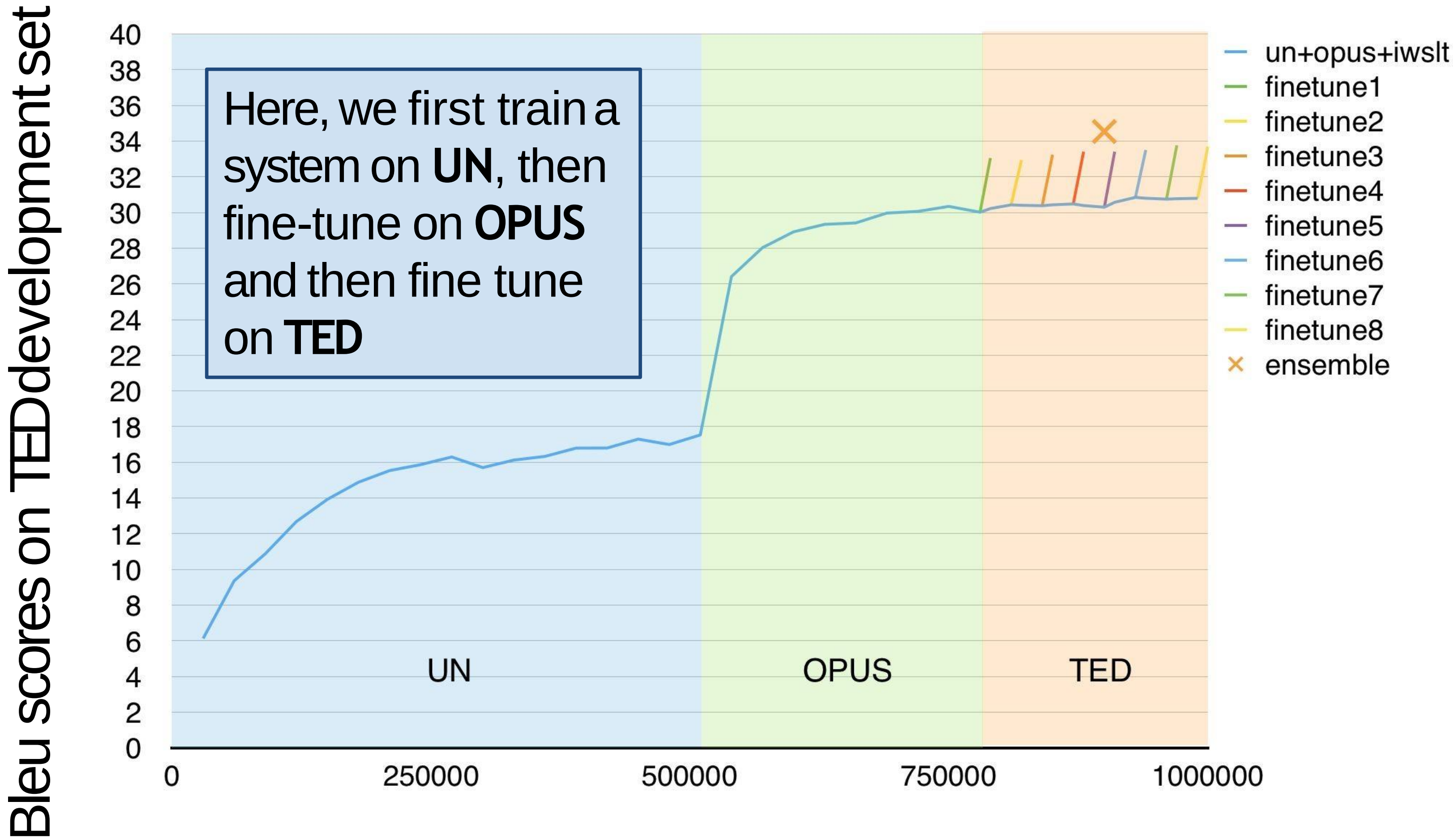
- Remember, NMT runs in mini-batches. Essentially we are simply training an NMT system on a few more mini-batches but of TED corpus
- System learned initial parameters from a large out-of-domain corpora and later we fine-tuned the trained parameters in favor of our in-domain corpus

Domain Adaptation - Fine-tuning

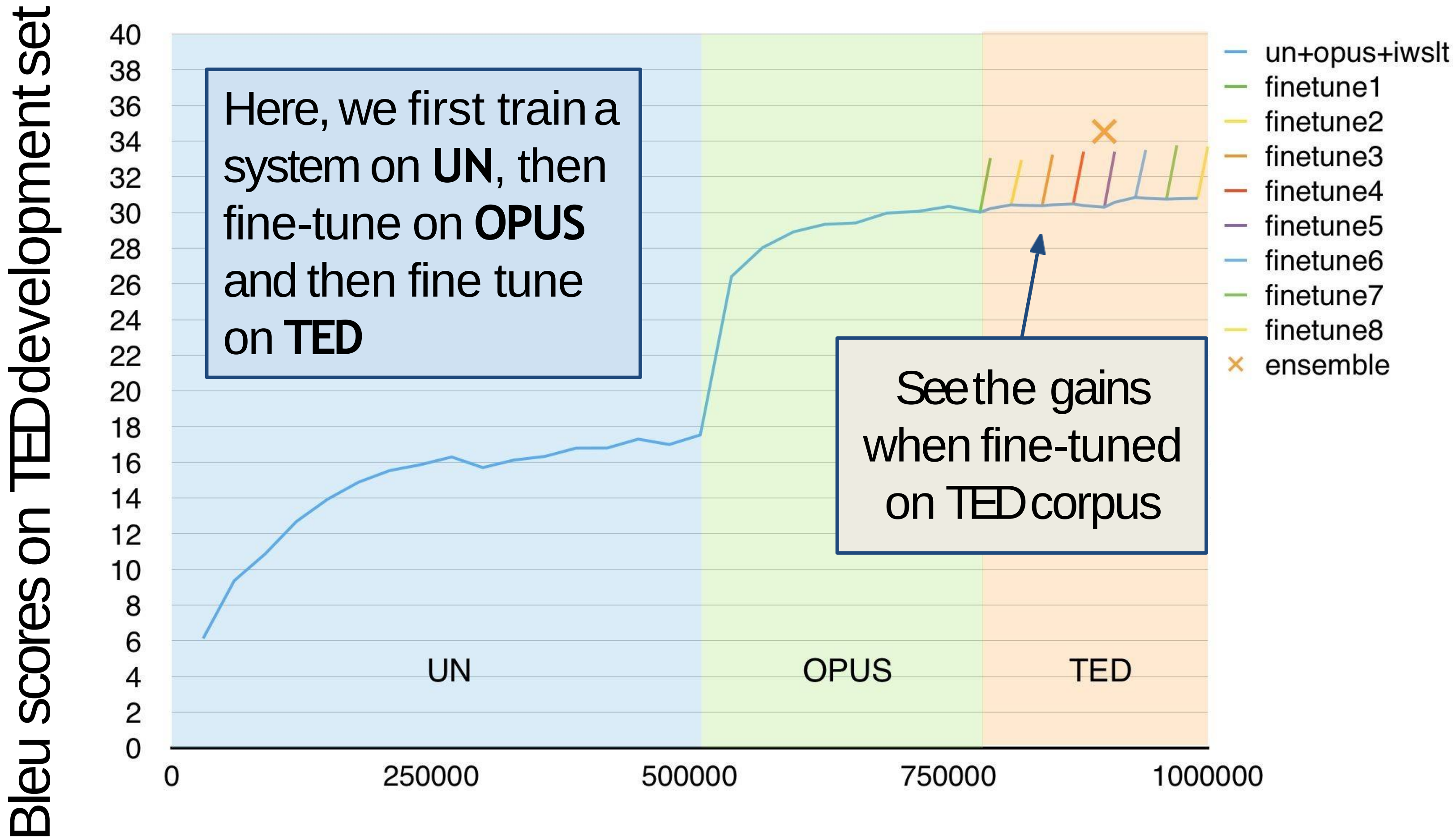


https://arxiv.org/pdf/1708.08712.pdf

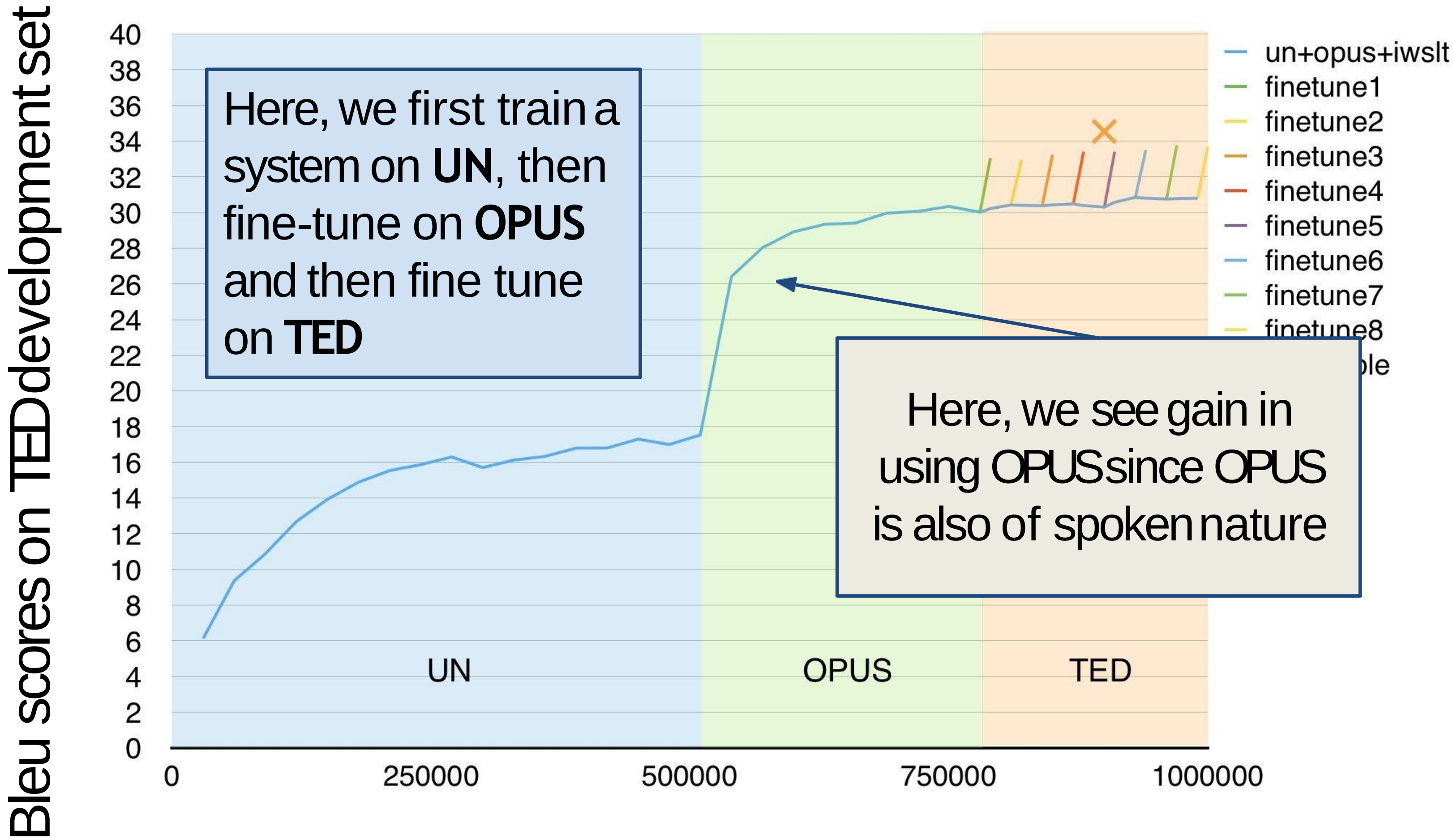
Domain Adaptation - Fine-tuning



Domain Adaptation - Fine-tuning



Domain Adaptation - Fine-tuning



Domain Adaptation

- Fine-tuning is an effective and commonly used method to bias model parameters towards a specific scenario
- Essentially build a general purpose model
- Fine-tune the model towards the domain of interest by resumming the training on the in-domain data

Multi-task Learning

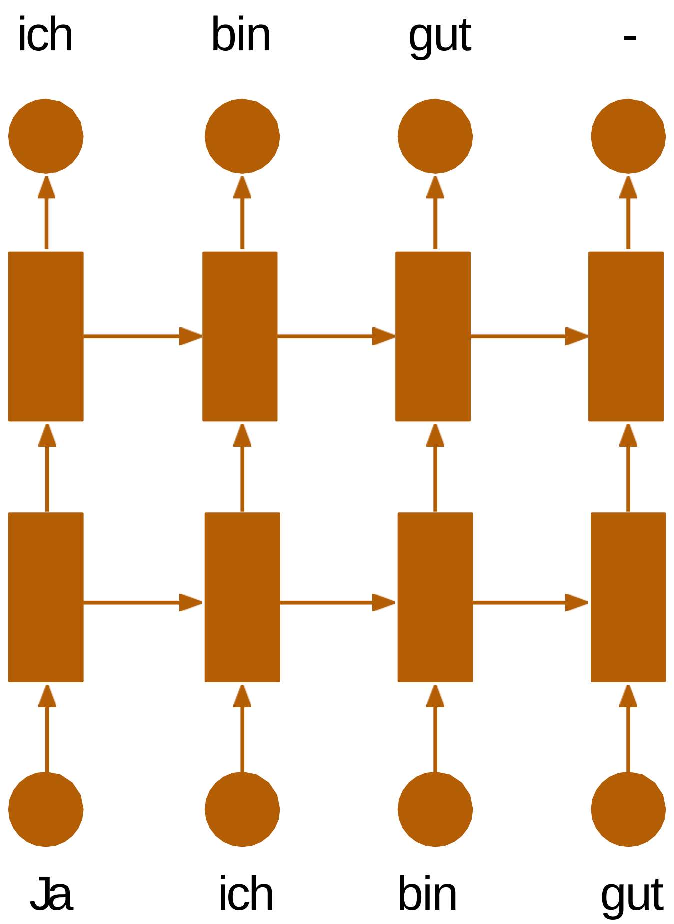
Multi-task Learning

- Learn multiple tasks together
  - Semantic tagging and POS tagging
  - Translation and POS tagging

- Implicitly introducing an inductive bias
  - Model has to consider more than one tasks to reduce loss
- Learning various tasks together helps to achieve better generalization

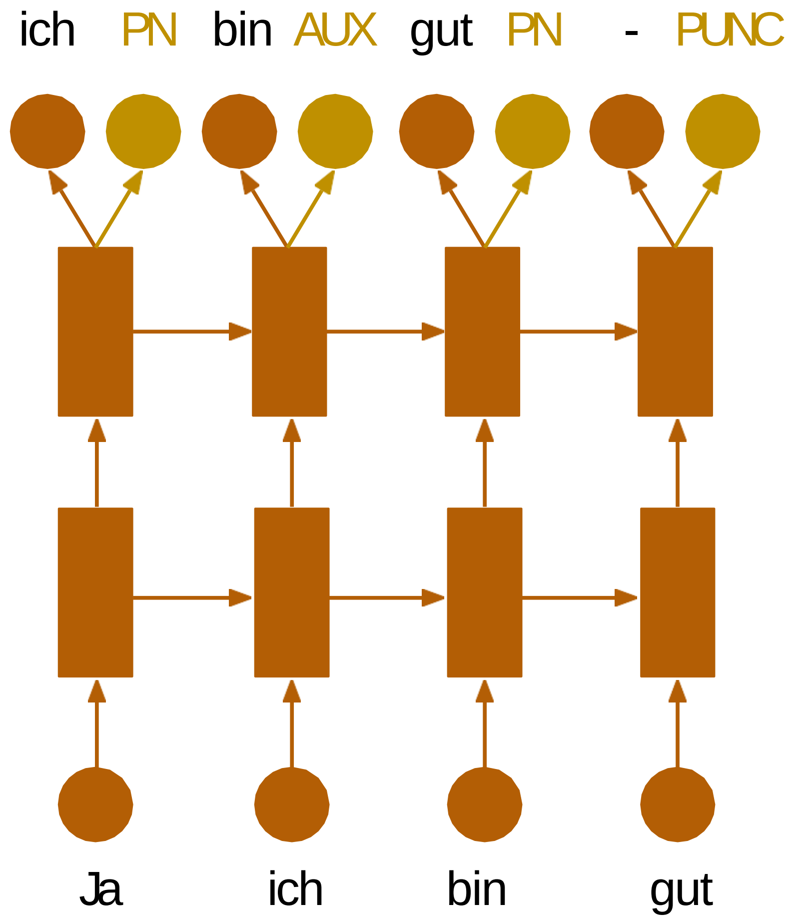
Multi-task Learning

Language Modeling with part of speech tagging



Multi-task Learning

Language Modeling with part of speech tagging



Multi-task Learning

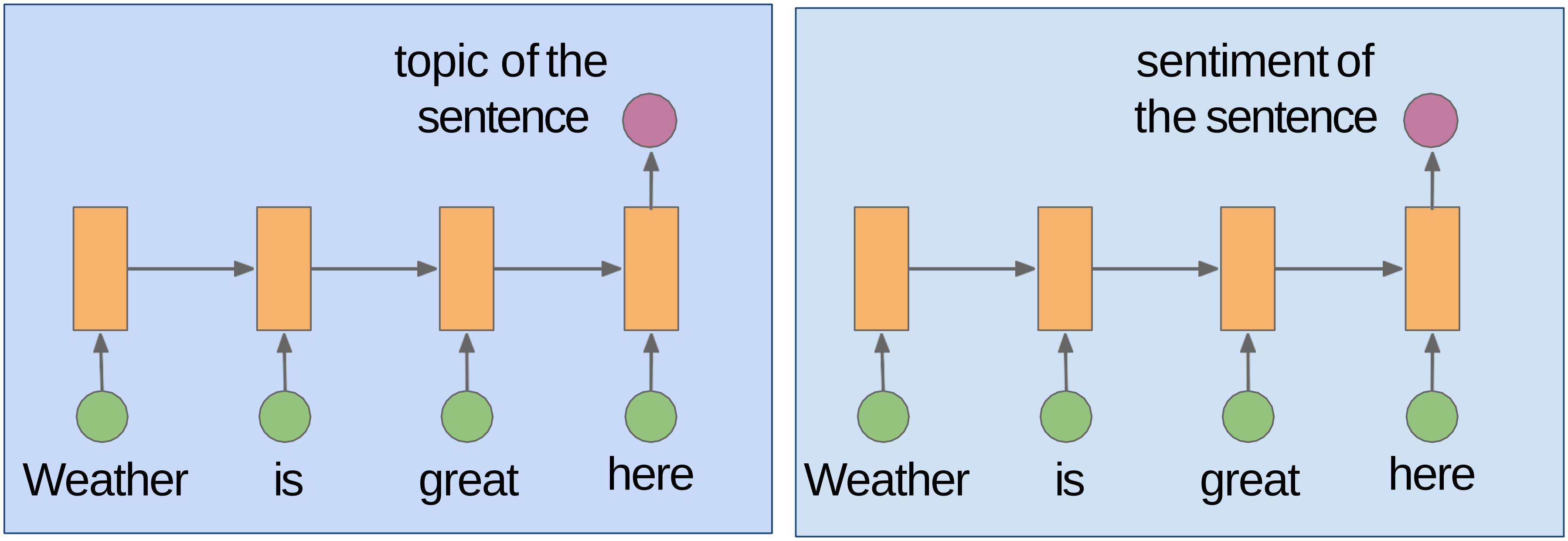
- Input format
  - Data: source sentence
  - Labels: target sentence, POS-tagged sequence
- Since we are learning weights of multiple tasks simultaneously, chances of overfitting are reduced
- As we increase the number of tasks, model has to find weights that minimize the overall loss
- This results in better generalization capability of the model

Multi-task Learning

- Let’s take another example to predict topic and sentiment of a sentence using an RNN

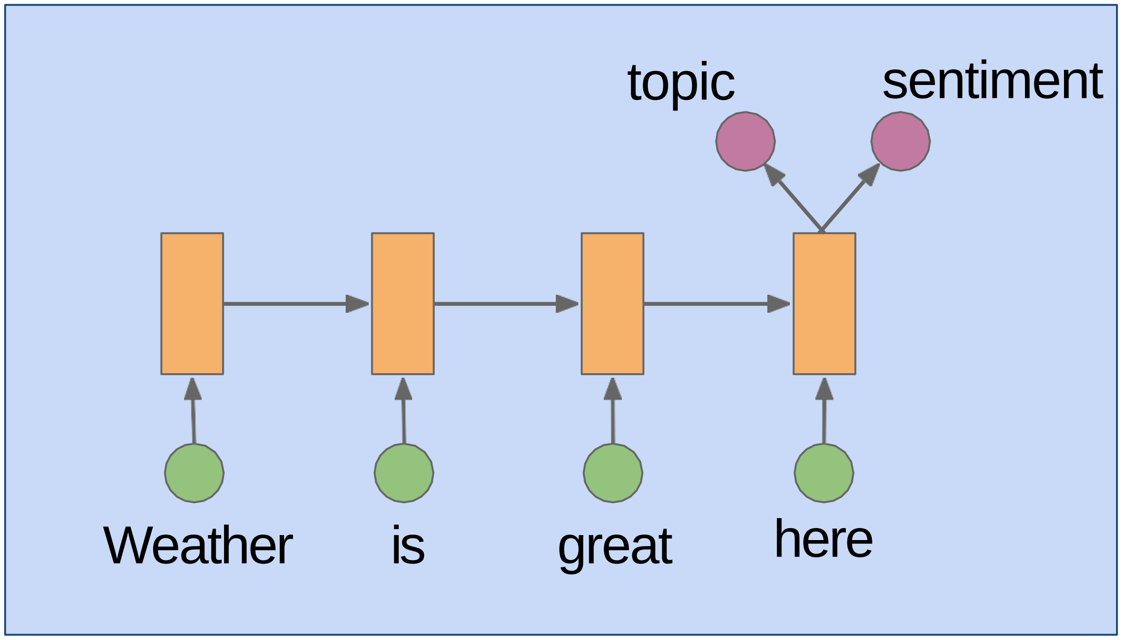
Multi-task Learning

- Let's take another example to predict topic and sentiment of a sentence using an RNN
- We have two models here



Multi-task Learning

- We can combine them into one model that predicts both topic and sentiment of a sentence



Multi-task Learning

- Calculate loss for each task
  - Loss of predicting topic and loss of predicting sentiment
  - Combine them as average loss, weighted average loss, etc.

- In essence, model learns to reduce the loss w.r.t to both tasks

Multi-task Learning

Various architectures to try:

- Shared embeddings
- Network of different sizes
- Various shared and unshared layers

Multi-source Multi-target

Multi-source Multi-target

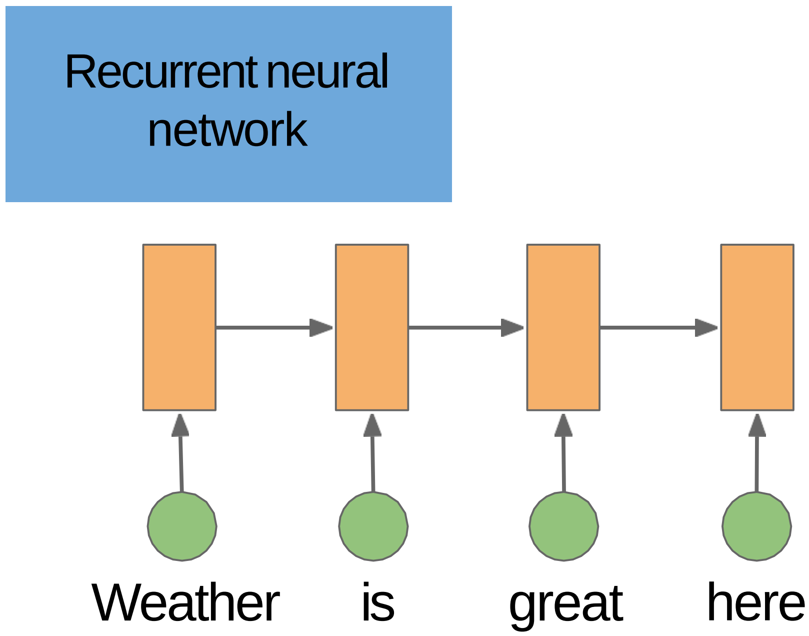
- In multi-task learning, we had one input with multiple output tasks
- One can build a model that takes one or more inputs and learns one or more output tasks
- For example, learn to predict POS tags of related languages together
  - The shared information across languages help to learn a better model
- Another example, learn translation between several closely related languages together

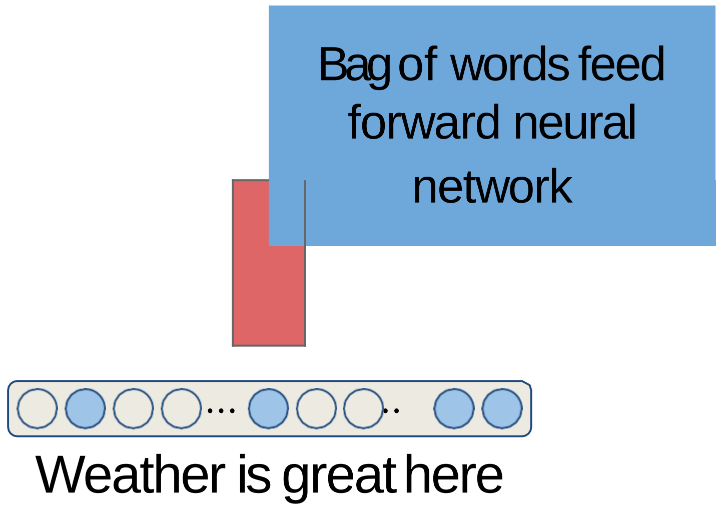
Multi-source Single-target

Let’s first look at multi-source single-target model

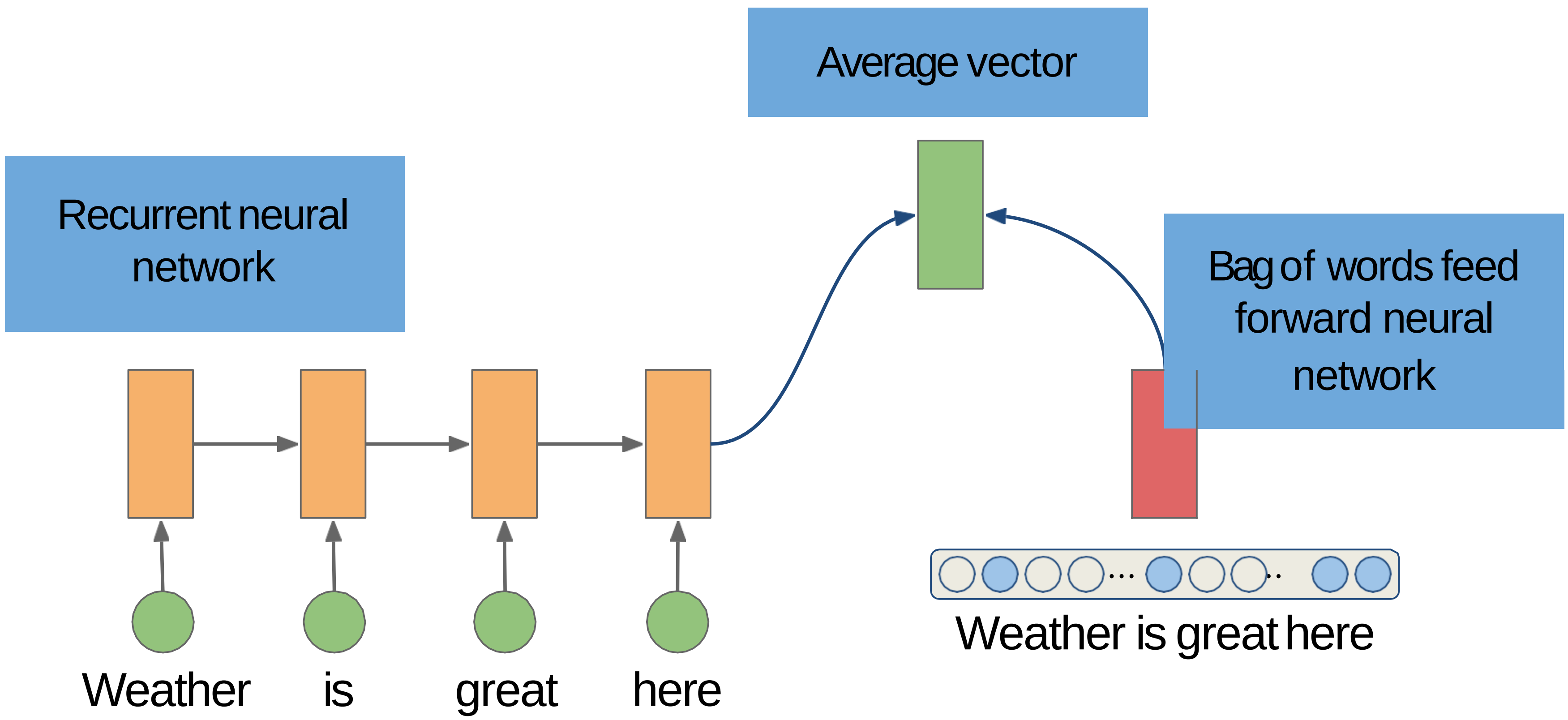
- Sequence classification task
- We have two networks
  - Feed forward neural network with bag of words as input
  - Recurrent neural network with word sequence as input
- We can combine both networks to perform the classification task

Multi-source Single-target

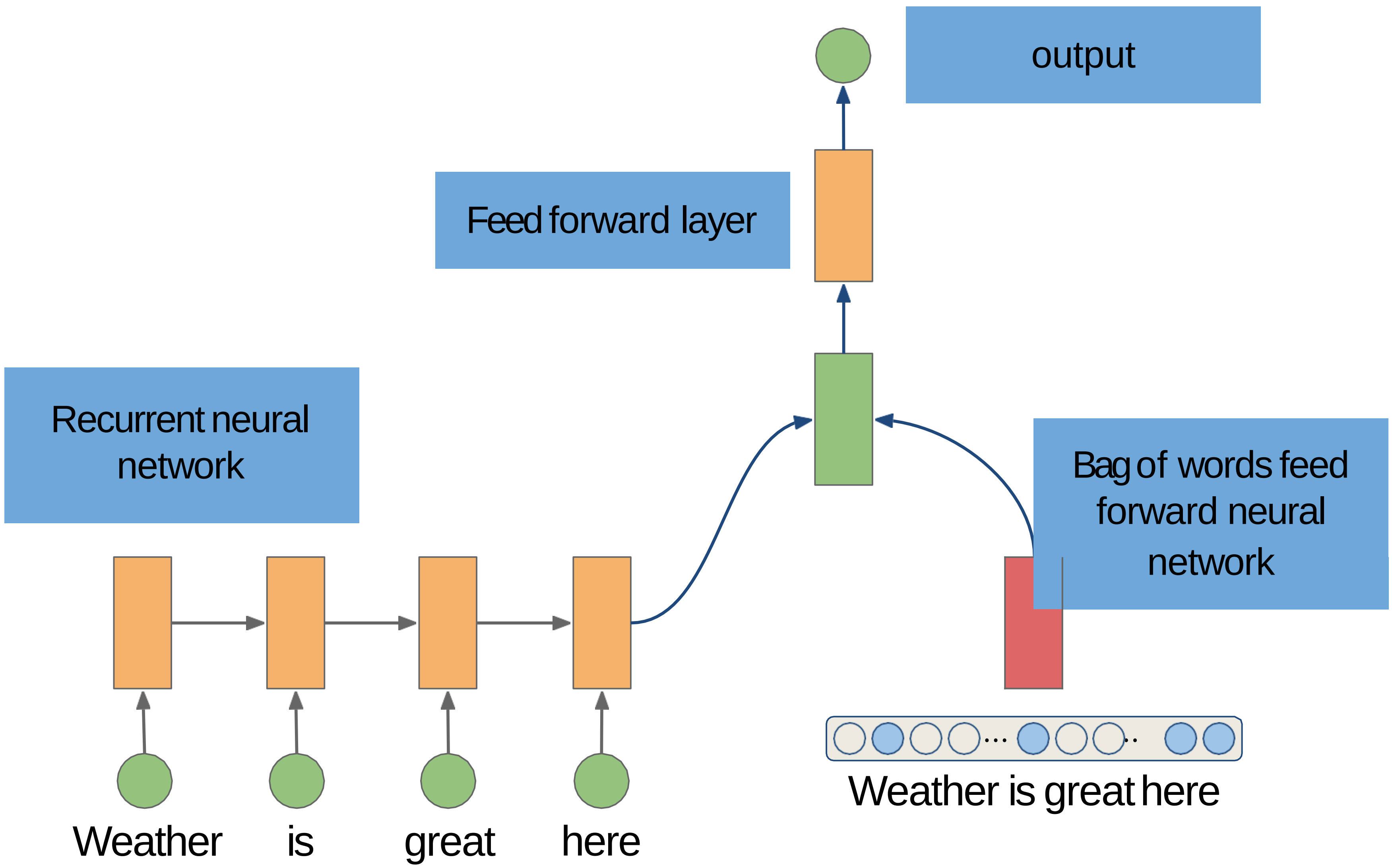




Multi-source Single-target



Multi-source Single-target



Multi-source Single-target

- The network has a share layer that combines the benefit of both inputs

Multi-source Multi-target

- We can also build networks that take multiple inputs and perform multiple tasks
- Inputs and outputs can be of different types
- These systems benefit from the data of several tasks
- Google NMT is one example of multi-lingual system that is trained on multiple source and target languages

Multi-modal Learning

Multi-modal Learning

Humans tend to learn using various senses e.g. babies use vision and speech together to learn communication

Multi-modal Learning

Humans tend to learn using various senses e.g. babies use vision and speech together to learn communication

We have seen the we can use the same encoder or decoder with more than one language

Multi-modal Learning

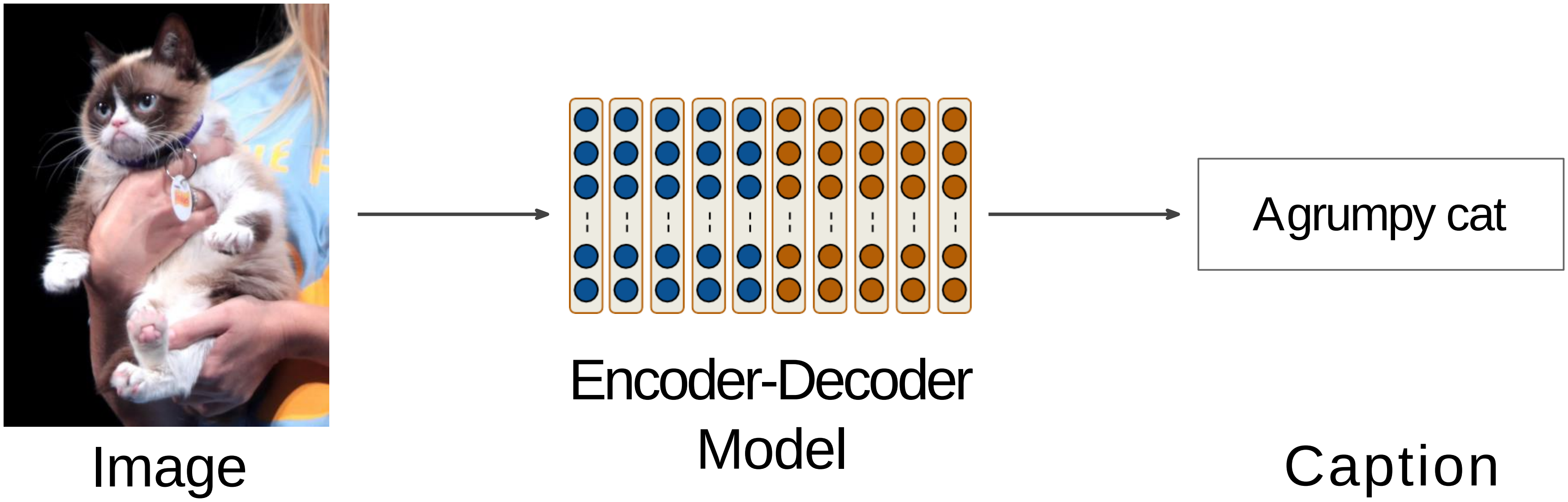
Q: As there was no inherent limitation of a single language for the encoder/decoder, is there any inherent limitation for the source/target to be a language at all?

Multi-modal Learning

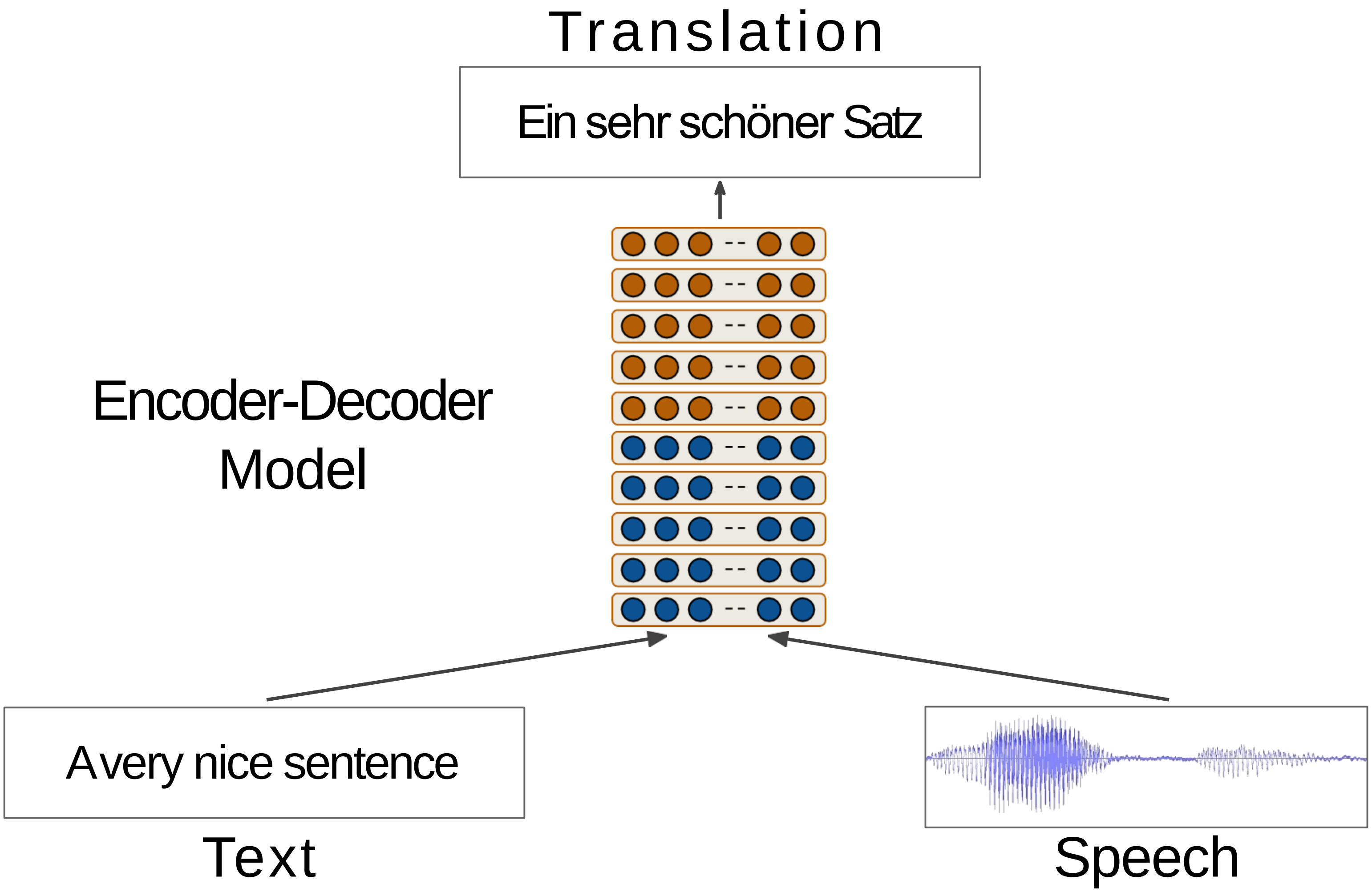
Q: As there was no inherent limitation of a single language for the encoder/decoder, is there any inherent limitation for the source/target to be a language at all?

A: No! As long as we can teach the machine to convert from any arbitrary input to its intermediate language, we should be able to work with that input!

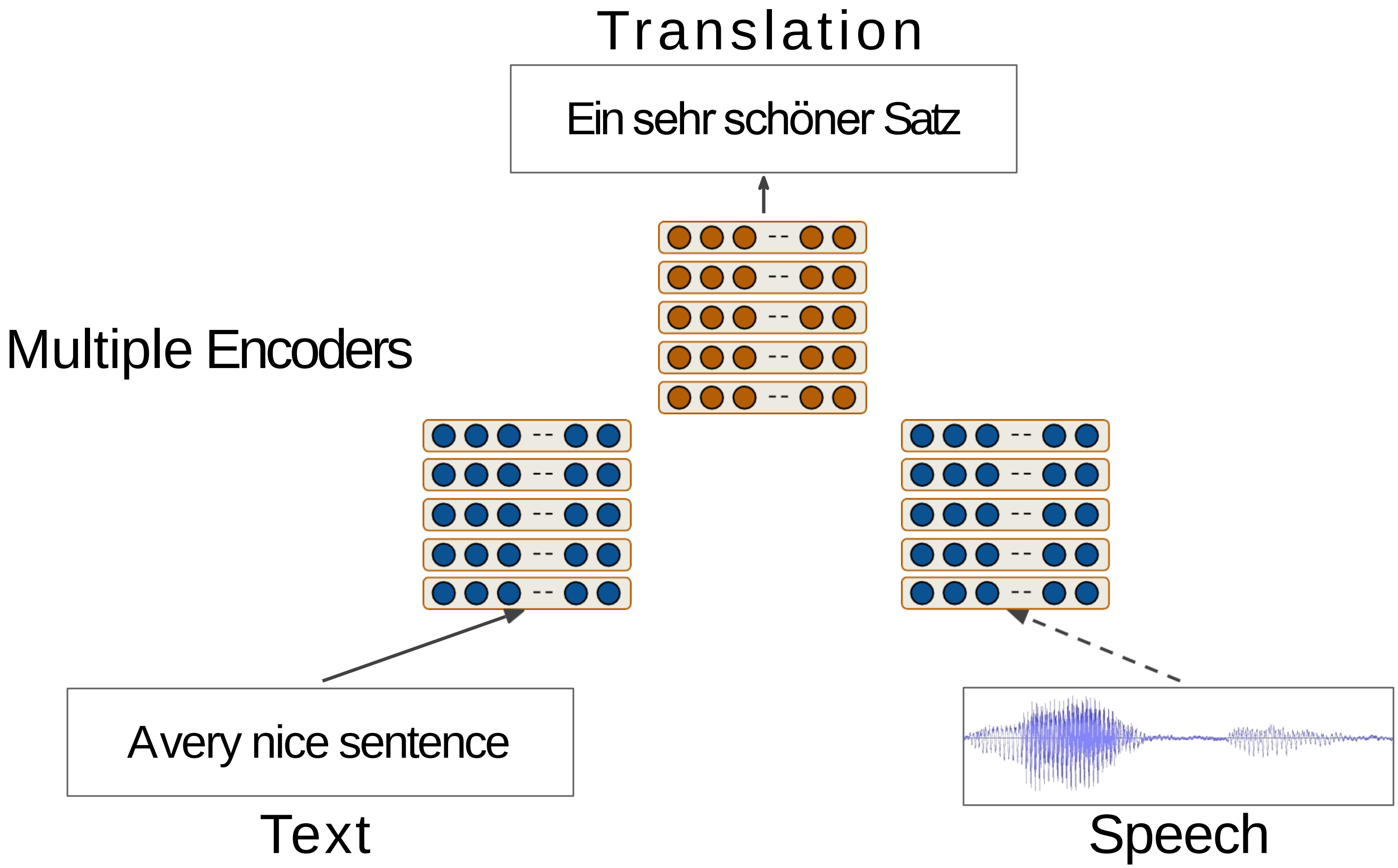
Multi-modal Learning



Multi-modal Learning



Multi-modal Learning



Multimodal Multitask Learning

Each encoder is usually designed for a specific type of input (text, speech, images)

The challenging part is to bring all senses to one space. Jointly training all these tasks with combined losses is one way to do this - still an open problem!

Recent Advancements

Generative Adversarial Networks

Generative Adversarial Networks

So far, we have seen only one class of machine learning problems: Classification problems

Generative Adversarial Networks

So far, we have seen only one class of machine learning problems: Classification problems

Another class of problems is generating outputs - say generating speech, drawing pictures, producing “handwritten” texts...

Generative Adversarial Networks

We can do this using **Generative Adversarial Networks**!

Humans have evolved over time, and a fundamental motivator for a lot of the progress has been *competition*

Generative Adversarial Networks

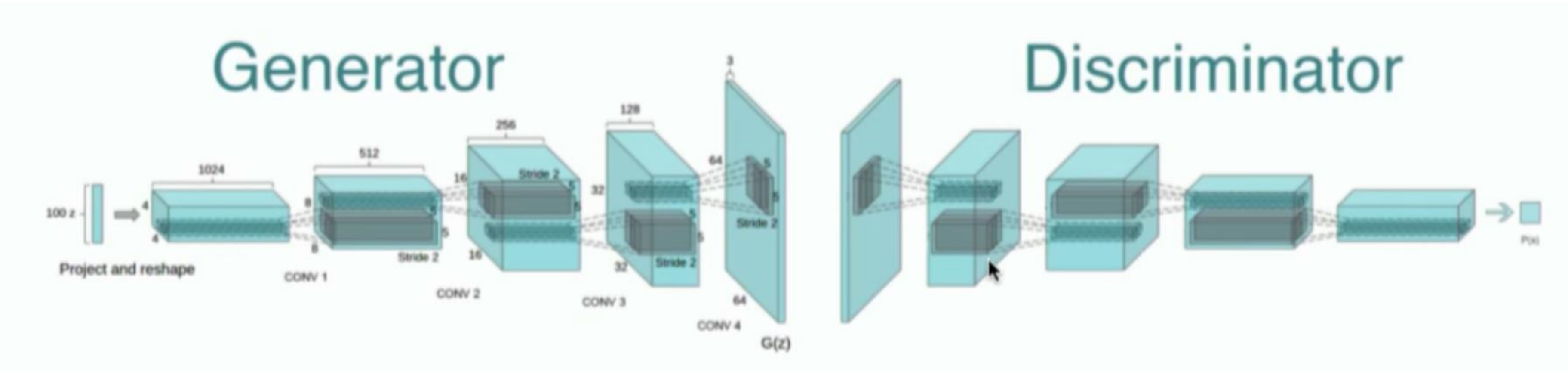
We can do this using **Generative Adversarial Networks**!

Humans have evolved over time, and a fundamental motivator for a lot of the progress has been *competition*

Idea: If humans (generally) get better because of *competition*, why not make machines compete with each other as well!

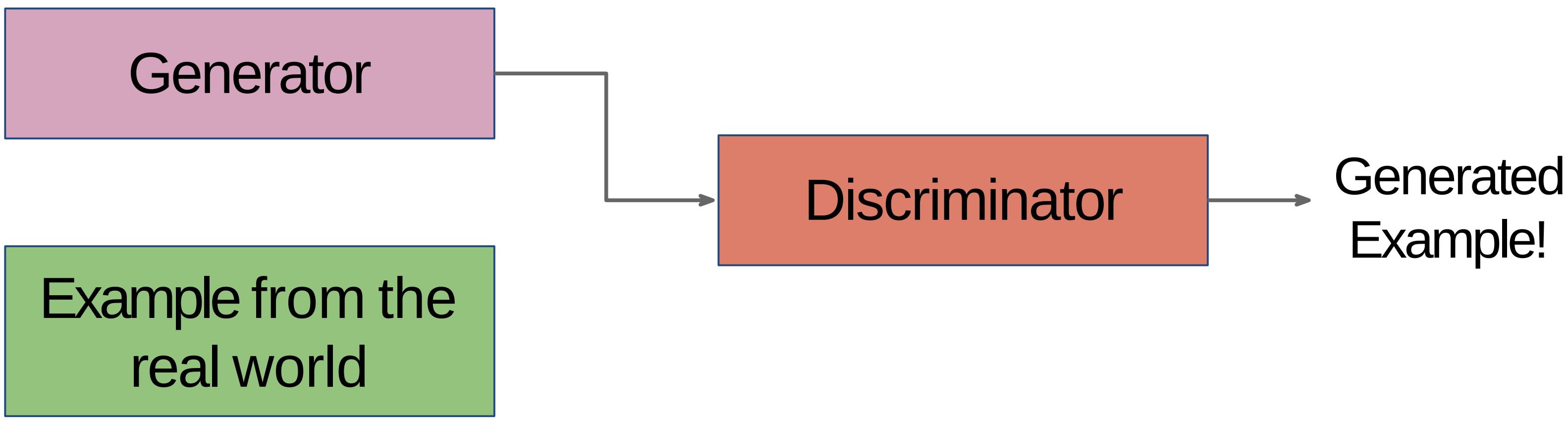
Generative Adversarial Networks

Intuition: We will have two neural networks - one will try and generate something, while the other will try to distinguish if its input is from the real world, or is generated by the first neural network!



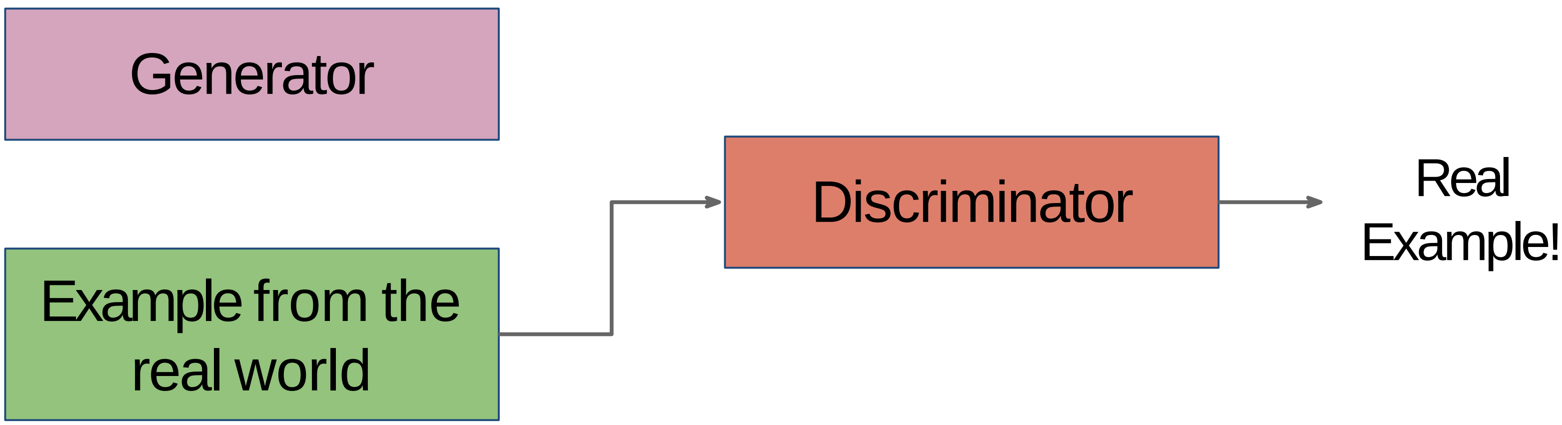
Generative Adversarial Networks

Intuition: We will have two neural networks - one will try and generate something, while the other will try to distinguish if its input is from the real world, or is generated by the first neural network!



Generative Adversarial Networks

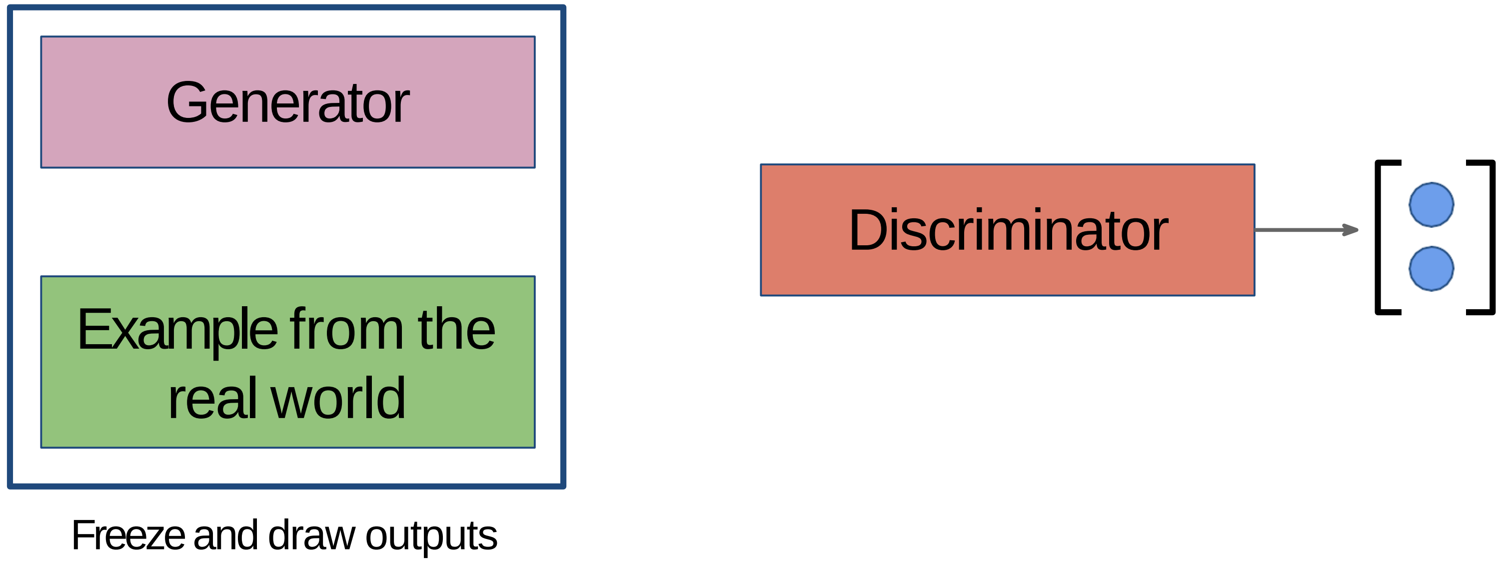
Intuition: We will have two neural networks - one will try and generate something, while the other will try to distinguish if its input is from the real world, or is generated by the first neural network!



Generative Adversarial Networks

Training process:

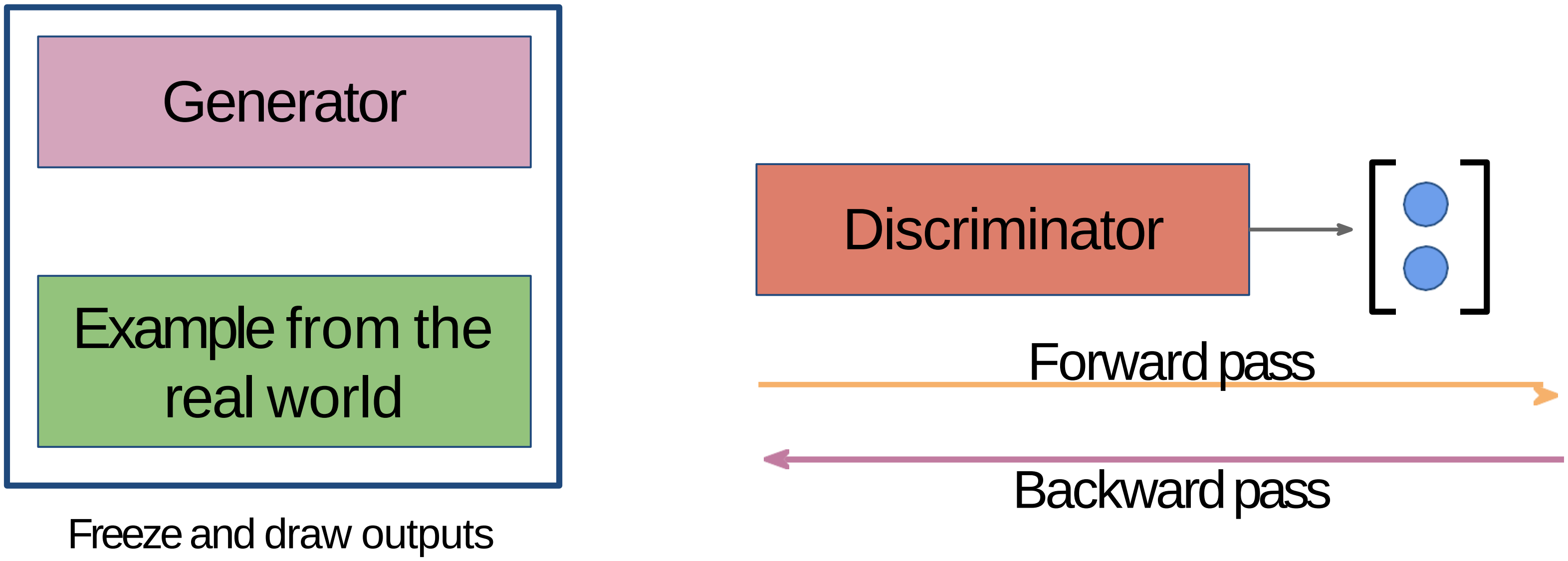
1. Freeze Generator and draw outputs from it
2. Draw equal number of outputs from the real world
3. Train the Discriminator on the mix of this data



Generative Adversarial Networks

Training process:

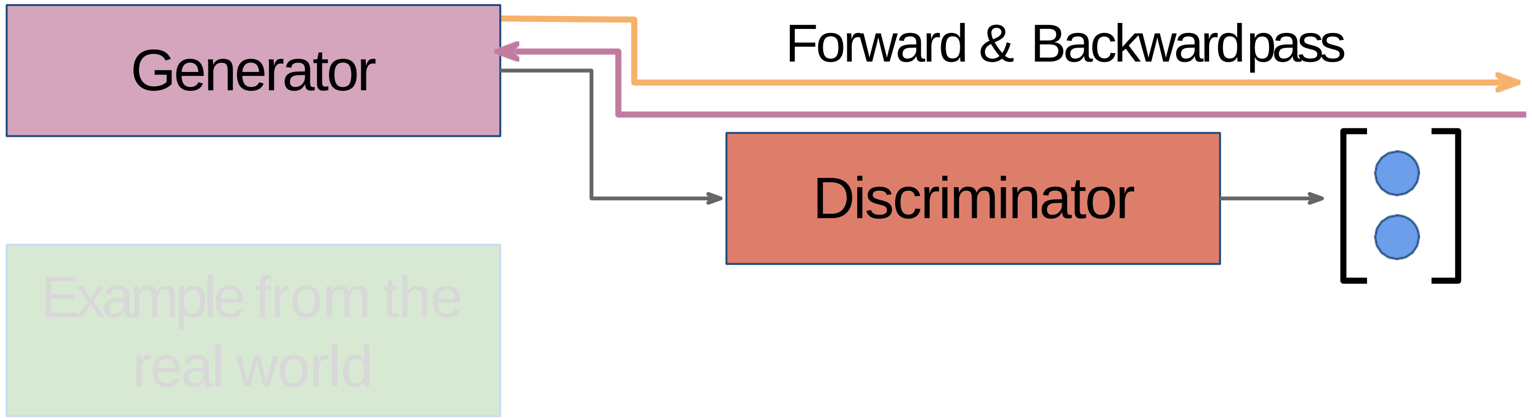
1. Freeze Generator and draw outputs from it
2. Draw equal number of outputs from the real world
3. Train the Discriminator on the mix of this data



Generative Adversarial Networks

Training process:

4. Freeze Discriminator
5. Draw an output from the generator
6. Pass the output through the frozen discriminator, and backpropagate the loss!

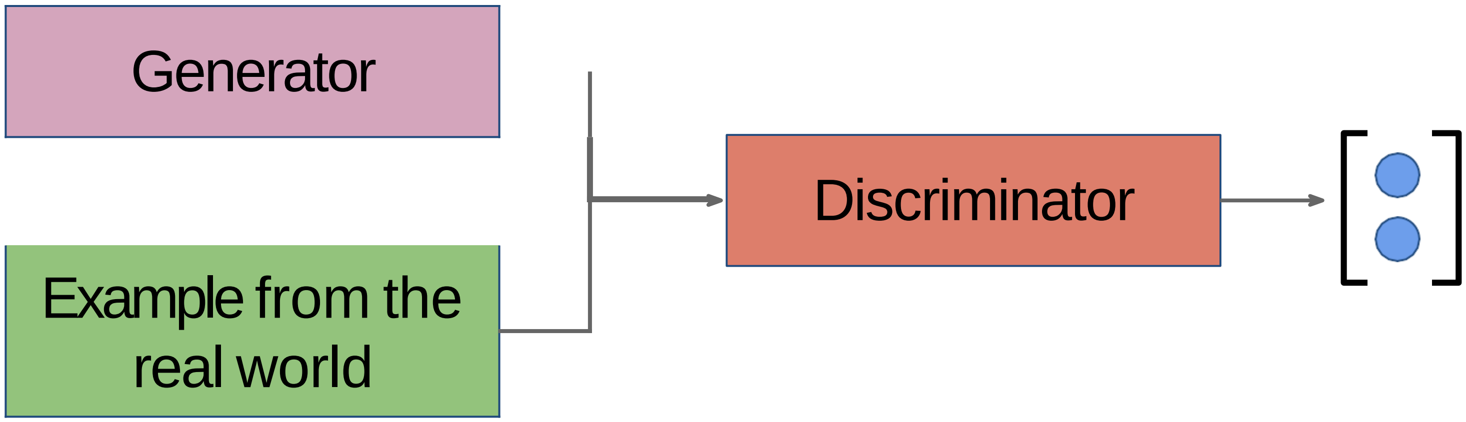


Generative Adversarial Networks

Training process:

7. Rinse and Repeat steps 1-6

8. Use the generator to generate real-world like examples!



Generative Adversarial Networks

https://www.reddit.com/r/MachineLearning/comments/49wrt4/adversarial_images_for_deep_learning/

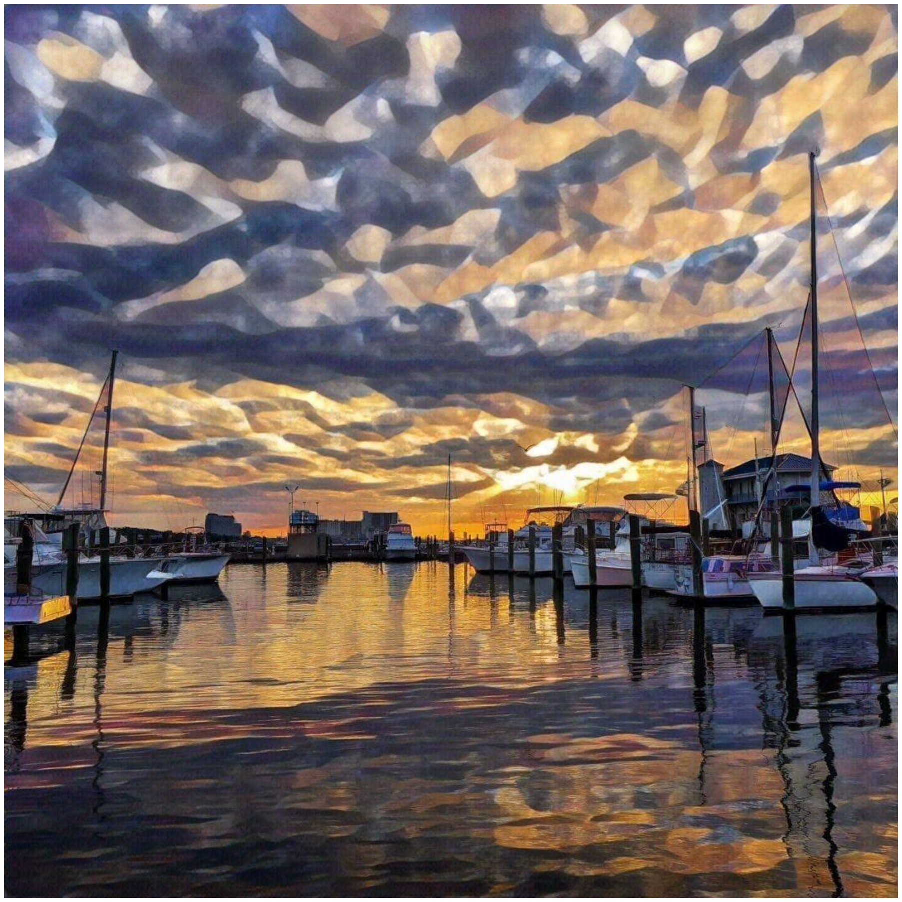
Generative Adversarial Networks

Learning what bedrooms look like:

“Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”
https://arxiv.org/abs/1511.06434v2

Generative Adversarial Networks

Prisma App: Art from pictures!



https://twitter.com/lapdog99/status/760639333812490240

Recent Advancements

Reinforcement Learning

Reinforcement Learning

Another class of problems is teaching a machine to perform some **sequence of actions** to **reach an eventual goal**!

Reinforcement Learning

Another class of problems is teaching a machine to perform some **sequence of actions** to **reach an eventual goal**!

Imagine a child's mind when it's learning to walk:

1. The child would notice how adults around it walk
2. It will try to first stand and balance itself - and will fall repeatedly before finding the right "parameters"
3. It will then learn to take small steps. Again, tuning its "parameters" to avoid falling

Reinforcement Learning

This is kind of what reinforcement learning algorithms do!

The task at hand is to train an “agent” who will perform some “actions” and eventually either succeed or fail.

Reinforcement Learning

This is kind of what reinforcement learning algorithms do!

The task at hand is to train an “agent” who will perform some “actions” and eventually either succeed or fail.

Researchers have come up with nice algorithms so that the “agent” can learn from its failures - and eventually succeed

Reinforcement Learning

But first, an example!

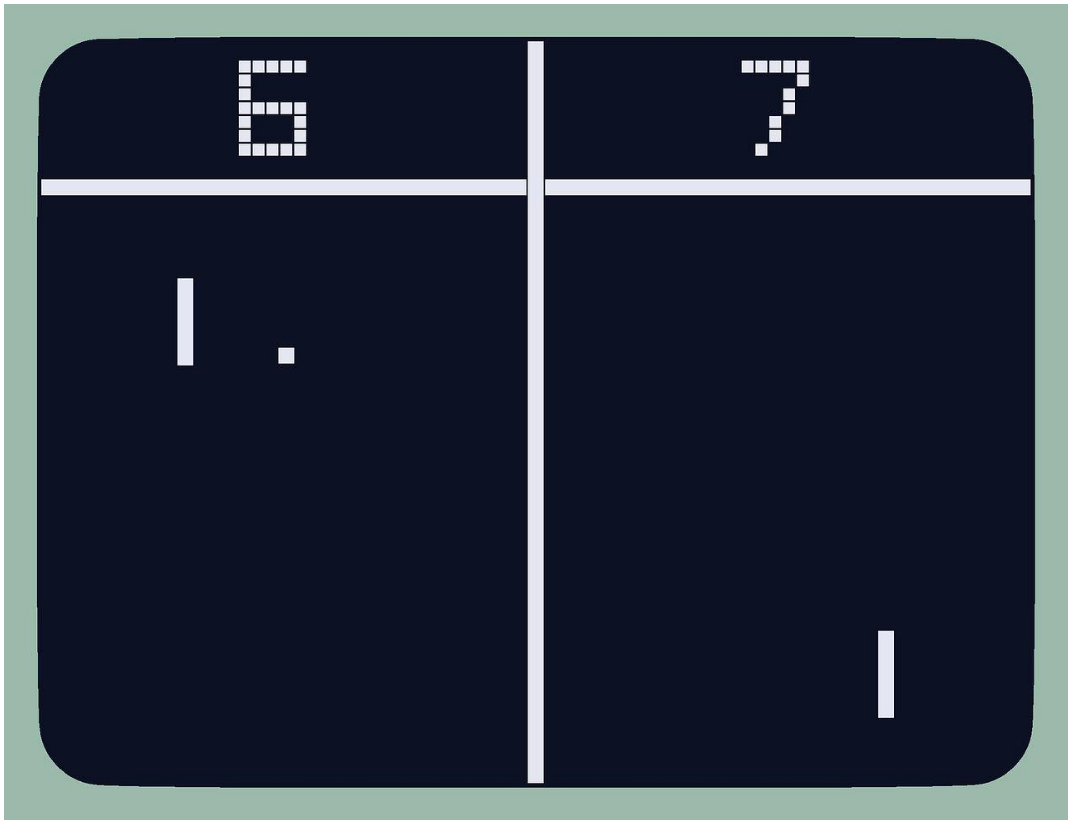
http://heli.stanford.edu/

Reinforcement Learning

A simplified walkthrough: Playing Pong!

Reinforcement Learning

Goal: We want to teach our computer to play the game of Pong



Reinforcement Learning

Goal: We want to teach our computer to play the game of Pong

General Idea: At each timestep (say every 100ms), we want to see where the ball is, and decide if we want to move our paddle up or down

Reinforcement Learning

Training process:

1. Initially, we will move our paddle up or down randomly, and most probably fail

Reinforcement Learning

Training process:

1. Initially, we will move our paddle up or down randomly, and most probably fail
2. Before failing, we have taken a set of actions

[up, down, up, up, up, down, down, up, down …]

Reinforcement Learning

Training process:

1. Initially, we will move our paddle up or down randomly, and most probably fail
2. Before failing, we have taken a set of actions

[up, down, up, up, up, down, down, up, down …]

3. Some action(s) caused us to eventually fail, so let us penalize all these actions a little bit (In some variations, penalize recent actions more)

Reinforcement Learning

Training process:

1. Initially, we will move our paddle up or down randomly, and most probably fail
2. Before failing, we have taken a set of actions

   [up, down, up, up, up, down, down, up, down ...]
3. Some action(s) caused us to eventually fail, so let us penalize all these actions a little bit (In some variations, penalize recent actions more)
4. Every time we succeed, boost all the actions a little bit

Reinforcement Learning

Training process:

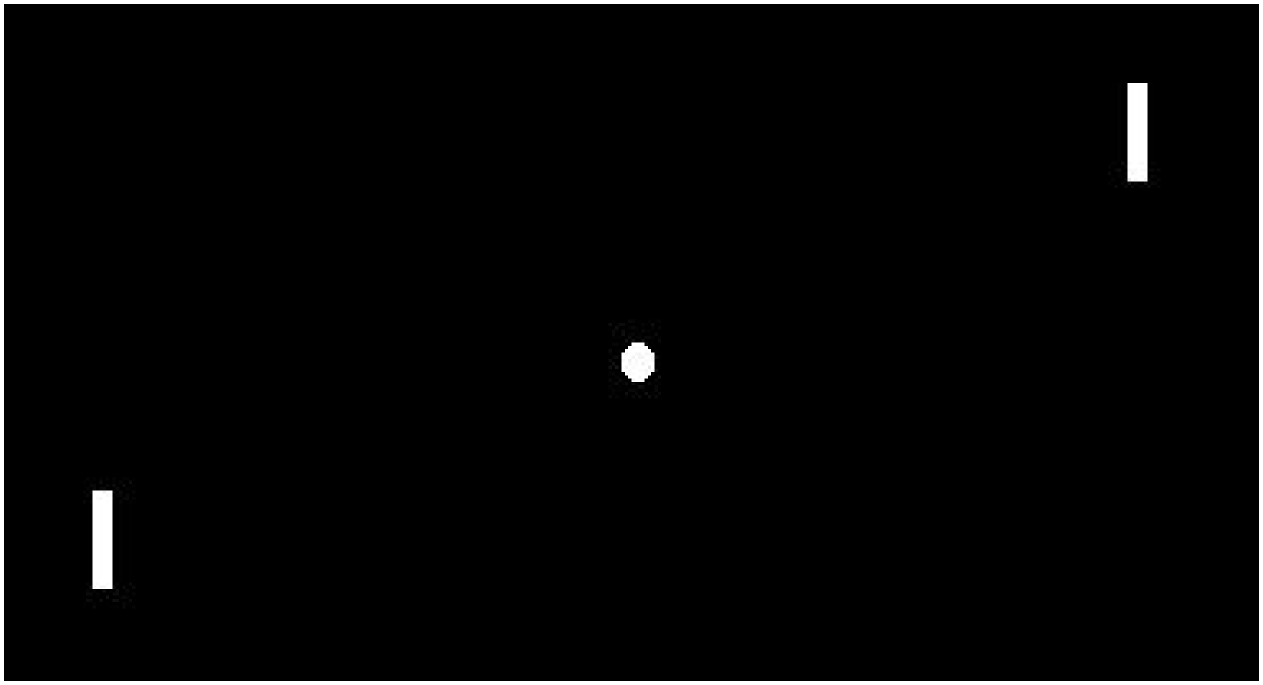
1. Initially, we will move our paddle up or down randomly, and most probably fail
2. Before failing, we have taken a set of actions

   [up, down, up, up, up, down, down, up, down …]
3. Some action(s) caused us to eventually fail, so let us penalize all these actions a little bit (In some variations, penalize recent actions more)
4. Every time we succeed, boost all the actions a little bit
5. Over time, "good actions at the right timesteps" will be boosted, and incorrect actions will be penalized!

Reinforcement Learning

Now make your computer play a few million games of Pong.

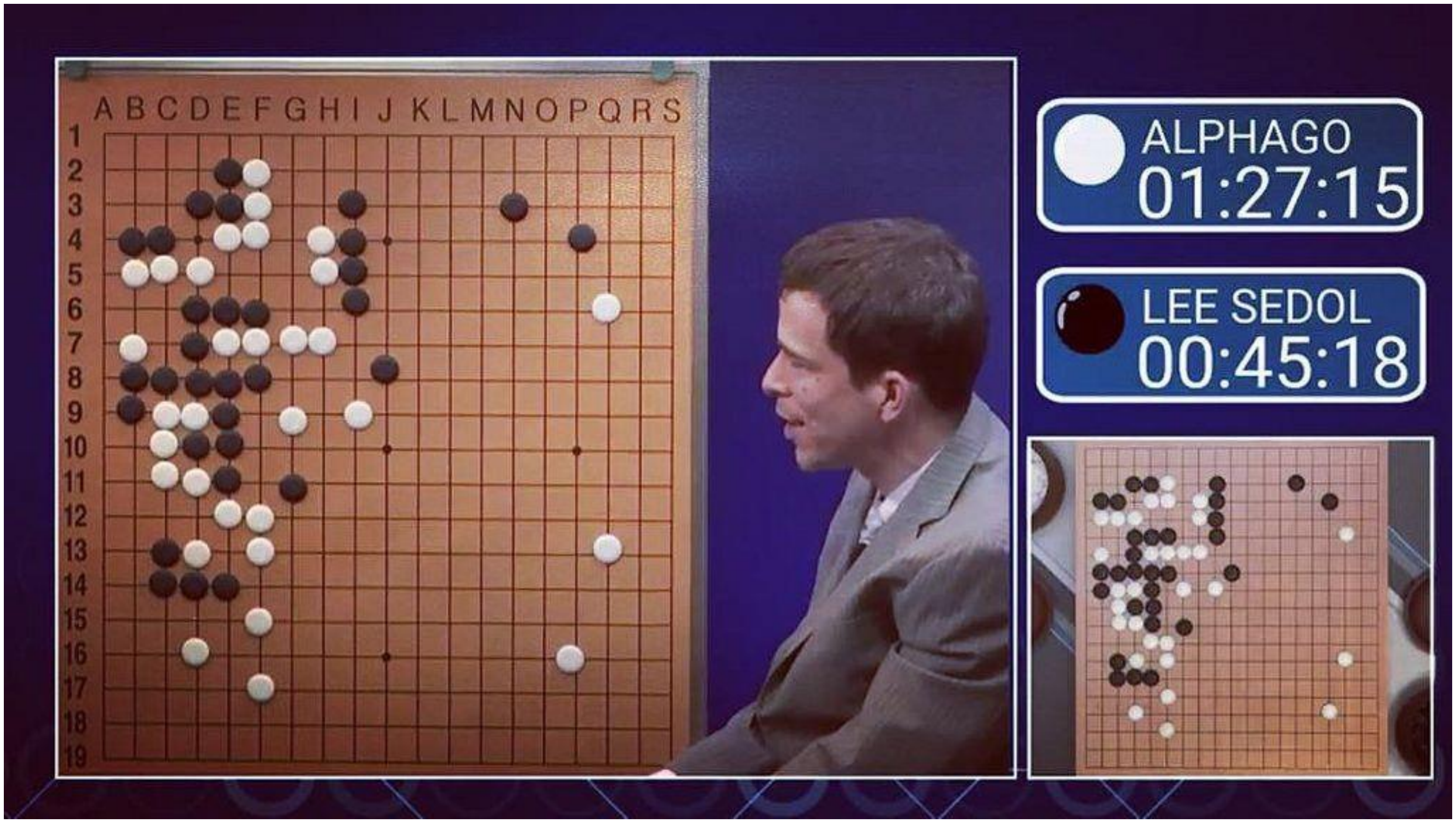
Overtime, it will learn to perform the right actions depending on the location of the ball!



http://karpathy.github.io/2016/05/31/rl/

Reinforcement Learning

More examples: AlphaGo



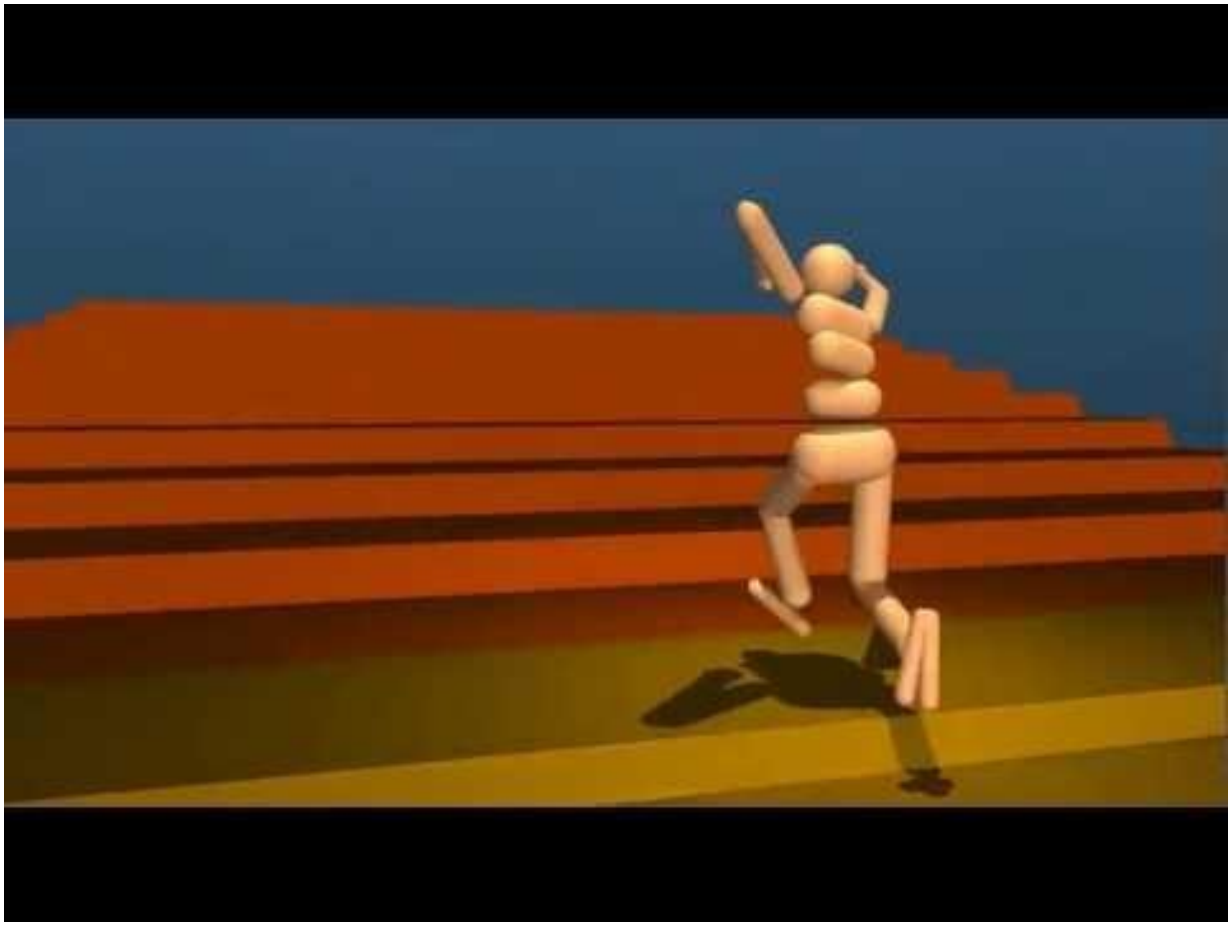
Reinforcement Learning

More examples: DOTA

Reinforcement Learning

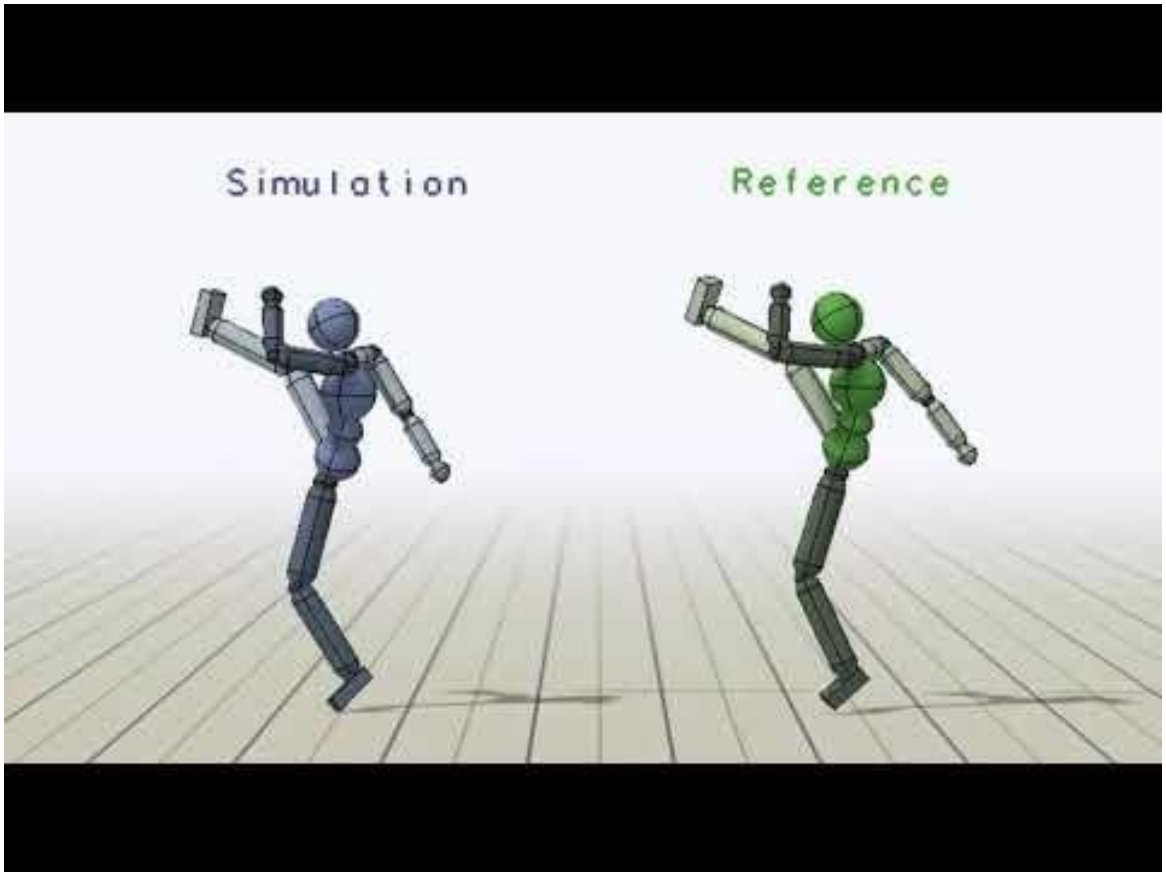
https://www.youtube.com/watch?v=faDKMMwOS2Q

More examples: Not just games...



Reinforcement Learning

More examples: Not just games...



https://www.youtube.com/watch?v=XCLSkFKTWyg

Summary

- Domain adaptation aims to use all available data in favor of the in-domain data
- Multilingual systems enable Zero-shot translation
- Multi-task learning improves generalization capability of the model
- Multi-modal learning is the way forward to build general AI's that understand the world like humans do
- Generative Adversarial Networks aim to solve the inverse problem of generating instead of just classifying
- Reinforcement Learning is a super-general framework that can help us teach agents how to act and react in the real world