PyTorch and Practical Deep Learning

Shreya Shankar

CS329S Guest Lecture

January 27, 2021

My background

- BS and MS from Stanford Computer Science

My background

- BS and MS from Stanford Computer Science
  - Did research in adversarial machine learning

My background

- BS and MS from Stanford Computer Science
  - Did research in adversarial machine learning
  - Worked with gigabytes of data

My background

- BS and MS from Stanford Computer Science
  - Did research in adversarial machine learning
  - Worked with gigabytes of data
  - Trained one decision tree and hundreds of deep learning models in my AI classes

My background

- BS and MS from Stanford Computer Science
  - Did research in adversarial machine learning
  - Worked with gigabytes of data
  - Trained one decision tree and hundreds of deep learning models in my AI classes
- Joined an applied ML startup as the first ML engineer

My background

- BS and MS from Stanford Computer Science
  - Did research in adversarial machine learning
  - Worked with gigabytes of data
  - Trained one decision tree and hundreds of deep learning models in my AI classes
- Joined an applied ML startup as the first ML engineer
  - Worked with terabytes of data

My background

- BS and MS from Stanford Computer Science
  - Did research in adversarial machine learning
  - Worked with gigabytes of data
  - Trained one decision tree and hundreds of deep learning models in my AI classes
- Joined an applied ML startup as the first ML engineer
  - Worked with terabytes of data
  - Trained one deep learning model and thousands of decision trees at my job

Things you may already know

- 87% of data science projects don't make it to production[1]

[1] https://venturebeat.com/2019/07/19/why-do-87-of-data-science-projects-never-make-it-into-production/

Things you may already know

- 87% of data science projects don't make it to production[1]
- Data in the "real world" is not necessarily clean and balanced, like canonical benchmark datasets (ex: ImageNet)

[1] https://venturebeat.com/2019/07/19/why-do-87-of-data-science-projects-never-make-it-into-production/

Things you may already know

- 87% of data science projects don't make it to production[1]
- Data in the "real world" is not necessarily clean and balanced, like canonical benchmark datasets (ex: ImageNet)
- Data in the "real world" is always changing

[1] https://venturebeat.com/2019/07/19/why-do-87-of-data-science-projects-never-make-it-into-production/

Things you may already know

- 87% of data science projects don't make it to production[1]
- Data in the "real world" is not necessarily clean and balanced, like canonical benchmark datasets (ex: ImageNet)
- Data in the "real world" is always changing
- Showing high performance on a fixed train and validation set $\neq$ consistent high performance when that model is deployed

[1] `https://venturebeat.com/2019/07/19/why-do-87-of-data-science-projects-never-make-it-into-production/`

Modeling is not very sexy IRL

- What's cool?

Modeling is not very sexy IRL

- What's cool?
  - Fancy models in school, academia, and research

Modeling is not very sexy IRL

- What's cool?
  - Fancy models in school, academia, and research
  - Simple models consistently delivering returns in industry

Modeling is not very sexy IRL

- What's cool?
  - Fancy models in school, academia, and research
  - Simple models consistently delivering returns in industry
- In industry, most of the time is spent debugging and "maintaining" models post-deployment

Modeling is not very sexy IRL

- What's cool?
  - Fancy models in school, academia, and research
  - Simple models consistently delivering returns in industry
- In industry, most of the time is spent debugging and "maintaining" models post-deployment
- Bias towards simple models that get the job done

Modeling is not very sexy IRL

- What's cool?
    - Fancy models in school, academia, and research
    - Simple models consistently delivering returns in industry
- In industry, most of the time is spent debugging and "maintaining" models post-deployment
- Bias towards simple models that get the job done
    - Easy to debug while iterating in development

Modeling is not very sexy IRL

- What's cool?
  - Fancy models in school, academia, and research
  - Simple models consistently delivering returns in industry
- In industry, most of the time is spent debugging and "maintaining" models post-deployment
- Bias towards simple models that get the job done
  - Easy to debug while iterating in development
  - Easy to debug during live deployment

A modeling project I worked on

- Tried to identify home and workplace locations based on vehicle sensor data for drivers that elected to opt-in

A modeling project I worked on

- Tried to identify home and workplace locations based on vehicle sensor data for drivers that elected to opt-in
- Had lots of data, threw to deep learning unsupervised methods

A modeling project I worked on

- Tried to identify home and workplace locations based on vehicle sensor data for drivers that elected to opt-in
- Had lots of data, threw to deep learning unsupervised methods
- Tried a simpler clustering approach

A modeling project I worked on

- Tried to identify home and workplace locations based on vehicle sensor data for drivers that elected to opt-in
- Had lots of data, threw to deep learning unsupervised methods
- Tried a simpler clustering approach
- Best solution was just looking at most frequent location at 2AM

A modeling project I worked on

- Tried to identify home and workplace locations based on vehicle sensor data for drivers that elected to opt-in
- Had lots of data, threw to deep learning unsupervised methods
- Tried a simpler clustering approach
- Best solution was just looking at most frequent location at 2AM
- Smart feature engineering can trump all

Common non deep learning libraries

Prioritize fast iteration, collaboration, and ease of deployment and monitoring.
Wants: modularity, Python API, Spark or Hadoop flexibility

- `scikit-learn`

Common non deep learning libraries

Prioritize fast iteration, collaboration, and ease of deployment and monitoring.
Wants: modularity, Python API, Spark or Hadoop flexibility

- `scikit-learn`
- `xgboost`

Common non deep learning libraries

Prioritize fast iteration, collaboration, and ease of deployment and monitoring.
Wants: modularity, Python API, Spark or Hadoop flexibility

- `scikit-learn`
- `xgboost`
- `lightGBM`

Common non deep learning libraries

Prioritize fast iteration, collaboration, and ease of deployment and monitoring.
Wants: modularity, Python API, Spark or Hadoop flexibility

- `scikit-learn`
- `xgboost`
- `lightGBM`
- `statsmodels`

Common non deep learning libraries

Prioritize fast iteration, collaboration, and ease of deployment and monitoring.
Wants: modularity, Python API, Spark or Hadoop flexibility

- `scikit-learn`
- `xgboost`
- `lightGBM`
- `statsmodels`
- `MLLib`

But when to use deep learning in practice?

- The task can be reduced to a generally "solved" image or text task (ex: image recognition, segmentation, language models)

But when to use deep learning in practice?

- The task can be reduced to a generally "solved" image or text task (ex: image recognition, segmentation, language models)
- Ideally "fine-tuning" an existing model instead of training a model from scratch

But when to use deep learning in practice?

- The task can be reduced to a generally "solved" image or text task (ex: image recognition, segmentation, language models)
- Ideally "fine-tuning" an existing model instead of training a model from scratch
- Lots of data ($>$ 10k examples)

But when to use deep learning in practice?

- The task can be reduced to a generally "solved" image or text task (ex: image recognition, segmentation, language models)
- Ideally "fine-tuning" an existing model instead of training a model from scratch
- Lots of data ($>$ 10k examples)
- Data is approximately balanced (similar number of points per class or subgroup)

But when to use deep learning in practice?

- The task can be reduced to a generally "solved" image or text task (ex: image recognition, segmentation, language models)
- Ideally "fine-tuning" an existing model instead of training a model from scratch
- Lots of data ($>$ 10k examples)
- Data is approximately balanced (similar number of points per class or subgroup)
- Data coming in doesn't change too much over time

But when to use deep learning in practice?

Please do not contact me saying you achieved success with a big neural network *and* these criteria weren't met. Good for you. But to me, it is not worth the hassle to try and debug and monitor it in production if a simpler model can do the job.

Deep learning in practice

- Not talking about what deep learning is (take CS224n, CS230, or CS231n if you are interested)

Deep learning in practice

- Not talking about what deep learning is (take CS224n, CS230, or CS231n if you are interested)
- Talking about deep learning challenges, frameworks, and a small tutorial

Differences between industry and research needs

- Researchers want fast iteration on models, practitioners want "high performance" (low latency, good throughput)

Differences between industry and research needs

- Researchers want fast iteration on models, practitioners want "high performance" (low latency, good throughput)
- Researchers love Python, practitioners have to work with existing codebases (which don't usually have Python)

Differences between industry and research needs

- Researchers want fast iteration on models, practitioners want "high performance" (low latency, good throughput)
- Researchers love Python, practitioners have to work with existing codebases (which don't usually have Python)
- Practitioners deal with the post-deployment aftermath

Some challenges in maintaining deep learning models

- "Loud" problems

Some challenges in maintaining deep learning models

- "Loud" problems
  - New releases to the framework and deprecated ops

Some challenges in maintaining deep learning models

- "Loud" problems
  - New releases to the framework and deprecated ops
  - Too big for some hardware

Some challenges in maintaining deep learning models

- "Loud" problems
  - New releases to the framework and deprecated ops
  - Too big for some hardware
- "Quiet" problems

Some challenges in maintaining deep learning models

- "Loud" problems
  - New releases to the framework and deprecated ops
  - Too big for some hardware
- "Quiet" problems
  - DL model changes without informing downstream model dependencies

Some challenges in maintaining deep learning models

- "Loud" problems
  - New releases to the framework and deprecated ops
  - Too big for some hardware
- "Quiet" problems
  - DL model changes without informing downstream model dependencies
  - Metric performance drop (might seem sudden)

Deep learning frameworks

Most popular deep learning libraries among researchers now:[2]

- TensorFlow

In practice, when using deep learning, the goal is to quickly and easily spin up a working model, then maintain it as it is being used in production.

[2] https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominate

Deep learning frameworks

Most popular deep learning libraries among researchers now:[2]

- TensorFlow
- PyTorch

In practice, when using deep learning, the goal is to quickly and easily spin up a working model, then maintain it as it is being used in production.

[2] https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominate

Deep learning frameworks

Most popular deep learning libraries among researchers now:[2]

- TensorFlow
- PyTorch
- JAX

In practice, when using deep learning, the goal is to quickly and easily spin up a working model, then maintain it as it is being used in production.

[2] https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominate

My opinions on these deep learning frameworks

- The race for a "winning framework" is kind of stupid to follow[3]

[3] https://www.shreya-shankar.com/modeling-libraries/

My opinions on these deep learning frameworks

- The race for a "winning framework" is kind of stupid to follow[3]
- Modeling is just a tiny step in the machine learning pipeline

[3] https://www.shreya-shankar.com/modeling-libraries/

My opinions on these deep learning frameworks

- The race for a "winning framework" is kind of stupid to follow[3]
- Modeling is just a tiny step in the machine learning pipeline
- Multiple libraries can "win" if they each do something important

[3] https://www.shreya-shankar.com/modeling-libraries/

My opinions on these deep learning frameworks

- The race for a "winning framework" is kind of stupid to follow[3]
- Modeling is just a tiny step in the machine learning pipeline
- Multiple libraries can "win" if they each do something important
  - Ex: Fast model iteration is important! Eager frameworks do this well.

[3] https://www.shreya-shankar.com/modeling-libraries/

My opinions on these deep learning frameworks

- The race for a "winning framework" is kind of stupid to follow[3]
- Modeling is just a tiny step in the machine learning pipeline
- Multiple libraries can "win" if they each do something important
  - Ex: Fast model iteration is important! Eager frameworks do this well.
  - Ex: Monitoring in production is important! No one does this well.

[3] `https://www.shreya-shankar.com/modeling-libraries/`

My opinions on these deep learning frameworks

- The race for a "winning framework" is kind of stupid to follow[3]
- Modeling is just a tiny step in the machine learning pipeline
- Multiple libraries can "win" if they each do something important
  - Ex: Fast model iteration is important! Eager frameworks do this well.
  - Ex: Monitoring in production is important! No one does this well.
- For the sake of today's tutorial, we will use PyTorch

[3] https://www.shreya-shankar.com/modeling-libraries/

Problem setup

- For the sake of the tutorial, must find a problem that hits all criteria in slide 8

[4] https://huggingface.co/datasets/amazon_reviews_multi

Problem setup

- For the sake of the tutorial, must find a problem that hits all criteria in slide 8
- Task: 5-way sentiment classification on Amazon reviews dataset[4]

[4] `https://huggingface.co/datasets/amazon_reviews_multi`

Problem setup

- For the sake of the tutorial, must find a problem that hits all criteria in slide 8
- Task: 5-way sentiment classification on Amazon reviews dataset[4]
    1. Load dataset and see what is in it

[4] `https://huggingface.co/datasets/amazon_reviews_multi`

Problem setup

- For the sake of the tutorial, must find a problem that hits all criteria in slide 8
- Task: 5-way sentiment classification on Amazon reviews dataset[4]
  1. Load dataset and see what is in it
  2. Train a vanilla BERT model (painful, lots of code and engineering hacks)

[4] https://huggingface.co/datasets/amazon_reviews_multi

Problem setup

- For the sake of the tutorial, must find a problem that hits all criteria in slide 8
- Task: 5-way sentiment classification on Amazon reviews dataset[4]
    1. Load dataset and see what is in it
    2. Train a vanilla BERT model (painful, lots of code and engineering hacks)
    3. Use HuggingFace Trainer and fine-tune a pretrained BERT (much easier)

[4] https://huggingface.co/datasets/amazon_reviews_multi

Tutorial

https://colab.research.google.com/drive/1VafFxndq74pQaDKEjkOSqLdVrHxCP1gM?usp=sharing