Other Learning Topics

Dr. Pedro Casas
Senior Scientist
Data Science & Artificial Intelligence
AIT Austrian Institute of Technology @Vienna
Other Relevant Learning Topics

- NLP – Recurrent Neural Networks (RNNs) and More (Attention, Transformers)
- (Deep) Reinforcement Learning
- Generative Adversarial Networks (GANs)
- Graph Neural Networks (GNNs)
- Adversarial Learning
- Transfer Learning
- AutoML – Automated Machine Learning
Recurrent Neural Networks (RNNs)

- Not all problems can be converted into one with fixed-length inputs and outputs (e.g., speech, time-series, etc.)

- **Recurrent Neural Networks** (RNNs) are a family of neural networks to **process sequential data**

- **Recurrent**: the output is influenced not only by the actual input, but also by the history of inputs fed in the past, **allowing information to persist**

- Simplest RNN model – **vanilla RNN**

\[
h_t = \tanh(W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix})
\]
Recurrent Neural Networks (RNNs)

- RNNs usually **use the same weights across time**
- This simplifies the complexity of the model, making it easier to train

\[
\begin{align*}
    h_t &= \tanh \, W \left( \begin{array}{c} x_t \\ h_{t-1} \end{array} \right) \\
    y_t &= F(h_t) \\
    C_t &= \text{Loss}(y_t, \text{GT}_t)
\end{align*}
\]
Vanilla RNN – Long Term Dependence

- **Task:** word forecasting

- **Short-term** dependence: Bob is eating an apple.

- **Long-term** dependence (**context**): Bob likes apples. He is hungry and decided to have a snack. So now he is eating an apple.

- **In theory,** vanilla RNNs can handle arbitrarily long-term dependence.

- **In practice,** it’s difficult, due to vanishing gradient problem.
LSTMs are a more complex variation of an RNN that are able to learn long term dependencies.

They solve the issue with the vanishing gradient problem by replacing the simple update rule vanilla RNN with a gating mechanism.

The task of the gates is to control which information is important for the prediction at every time step and which is not.

Using this mechanism, a LSTM can decide what to remember and what to forget from the past to obtain better predictions.
Attention Mechanisms and Transformers

- The same problem that happens to RNNs generally, happen with LSTMs, i.e. **when sentences are too long LSTMs still don’t do too well** (context is lost)

- **Attention Mechanisms**: human attention allows us to focus on a certain elements/regions with “high resolution”, while perceiving the surrounding image in “low resolution”

- **Attention in deep learning** can be broadly interpreted as a **vector of importance weights**: to predict or infer one element, we estimate using the **attention vector** how strongly it is correlated with other elements

- Can therefore **provide sharper information about far-away elements**

She is eating a green apple.
Attention Mechanisms and Transformers

- When added to RNNs, attention mechanisms led to large gains in performance
  
  “Attention Is All You Need” (Vaswani, et al., NIPS 2017) → no need for RNNs!

- **Transformers**: a new model to handle sequential data, differently from RNNs

- Unlike RNNs, Transformers **do not require that the sequential data be processed in order**

- Therefore, it allows for much more parallelization than RNNs, **significantly reducing training times**, and to be trained more efficiently on **massive datasets**.

- Transformers are the default building block for NLP today
  
  - BERT (Google)
  - GPT (openAI)
  - etc...
Other Relevant Learning Topics

- NLP – Recurrent Neural Networks (RNNs) and More (Attention, Transformers)
- (Deep) Reinforcement Learning
- Generative Adversarial Networks (GANs)
- Graph Neural Networks (GNNs)
- Adversarial Learning
- Transfer Learning
- AutoML – Automated Machine Learning
Reinforcement Learning

- Reinforcement learning: **teach by experience**

**RL agent** may be directly or indirectly trying to learn a:

- **Policy**: agent’s behavior function – action selection (probability of taking action $a$ when in state $s$)

- **Value function**: how good is each state and/or policy (expected return starting in state $s$, following policy $\pi$)

- **Model**: agent’s representation of the environment
Reinforcement Learning – Exploration vs. Exploitation

- **Deterministic/greedy** (max. reward) policy won’t explore all actions
  - Don’t know anything about the environment at beginning
  - Need to try all actions to find the optimal one
  - With *incomplete knowledge of the world*, shall the agent **repeat decisions that have worked well** so far (**exploit**) or **make novel decisions** (**explore**), hoping to gain even greater rewards.

- **ε–greedy policy** to improve exploration-exploitation tradeoffs:
  - With probability 1-ε, perform the optimal/greedy action, otherwise random action
  - Slowly move it towards greedy policy: ε → 0
Reinforcement Learning

3 types of Reinforcement Learning:

- **Model-based**: learn the model of the environment, update it often, re-plan often
- **Value-based**: learn the state or state-action value (expected return)
- **Policy-based**: learn the stochastic policy function mapping state to action
Deep Reinforcement Learning

- Deep RL = RL + Deep Neural Networks

Networks for Learning Actions, Values, Policies, and/or Models

- Input: Environment Sample
- Network: Any Encoder
- Output: Action
- Experienced Ground Truth: Reward

Deep Reinforcement Learning

Deep Learning

Environment

Sensors

Sensor Data

Feature Extraction

Representation

Machine Learning

Knowledge

Reasoning

Planning

Action

Effector
Deep Reinforcement Learning Example – DOOM

**DOOM** - first-person shooter developed for MS-DOS

**Goal:**
eliminate all opponents

**State:**
raw image pixels of the game

**Actions:**
up, down, left, right, shoot, etc.

**Reward:**
positive when eliminating an opponent, negative when the agent is eliminated
Other Relevant Learning Topics

- NLP – Recurrent Neural Networks (RNNs) and More (Attention, Transformers)
- (Deep) Reinforcement Learning
- Generative Adversarial Networks (GANs)
- Graph Neural Networks (GNNs)
- Adversarial Learning
- Transfer Learning
- AutoML – Automated Machine Learning
Generative Models

- Given training data, **generate new samples from same distribution**

  ![Training data](image1) ![Generated samples](image2)

  \( \text{Training data } \sim p_{\text{data}}(x) \quad \text{Generated samples } \sim p_{\text{model}}(x) \)

- The **problem of generative models** is about **learning** \( p_{\text{model}}(x) \) **similar to** \( p_{\text{data}}(x) \)

- Generative model learning is about **density estimation**:
  - **Explicit density estimation**: explicitly define and solve for \( p_{\text{model}}(x) \)
  - **Implicit density estimation**: learn model that can sample from \( p_{\text{model}}(x) \) w/o explicitly defining it
Generative Adversarial Networks (GANs)

- Implicit density estimation through **game-theoretic approach**
- Learn to generate samples from training distribution through **2-players (minimax) game**
- **Problem:** want to sample from potentially complex, high-dimensional training distribution. No direct way to do this!
- **Solution:** sample from a simple distribution, e.g. **random noise**. Learn transformation to training distribution, **using a neural network**

- **Generator** network: tries to fool the discriminator by generating real-looking instances from random noise
- **Discriminator** network: tries to distinguish between real and fake instances

https://thispersondoesnotexist.com
Other Relevant Learning Topics

- NLP – Recurrent Neural Networks (RNNs) and More (Attention, Transformers)
- (Deep) Reinforcement Learning
- Generative Adversarial Networks (GANs)
- **Graph Neural Networks (GNNs)**
- Adversarial Learning
- Transfer Learning
- AutoML – Automated Machine Learning
Graph Neural Networks (GNNs)

- In a nutshell: **deep learning** architecture for **graph-structured data**

- Lots of domains where graph-structured data makes much more sense: social networks, knowledge graphs, recommender systems, **communication networks**

- Typical application of GNN: node classification → every node in the graph is associated with a label, and we want to predict the label of the nodes without ground-truth

- Have so far proved very powerful in modeling the dependencies between nodes in graph-like structures

- **About 4% of ICLR 2020 submitted papers using GNNs** (2585 submissions)

- **Graph Neural Networking Challenge 2020**, based on RouteNET: a GNN architecture to estimate per-source-destination performance metrics in communication networks
Other Relevant Learning Topics

- NLP – Recurrent Neural Networks (RNNs) and More (Attention, Transformers)
- (Deep) Reinforcement Learning
- Generative Adversarial Networks (GANs)
- Graph Neural Networks (GNNs)
- Adversarial Learning
- Transfer Learning
- AutoML – Automated Machine Learning
Adversarial Learning

- A technique which attempts to **fool models through malicious inputs** (aka adversarial examples)

- We saw that **concept drift** can make ML models to drastically fail. The same ideas behind concept drift can be exploited by an attacker to fool a ML model

- Strongly linked to the problem of **Robust ML** → building robust models

- **Adversarial Learning** shows how a malicious adversary can manipulate the input data to **exploit specific vulnerabilities of learning algorithms** and **compromise the security** of the full machine learning system
Adversarial Learning

- **Evasion attacks:**
  - executed at runtime of the system
  - evade detection by obfuscation, or even spoofing/synthetization
  - Exploratory nature

- **Poisoning attacks:**
  - executed at training time
  - It is about understanding the learning algorithm...
  - ...to introduce specific bias into the learning phase
Adversarial Learning – Examples

Fake-it through synthetization (e.g., GANs)
Adversarial Learning – Examples

Fool the CNN, not the human eye
Adversarial Learning – Examples

Adversarial Glasses
Adversarial Learning – Examples

Adversarial Traffic Signs
Other Relevant Learning Topics

- NLP – Recurrent Neural Networks (RNNs) and More (Attention, Transformers)
- (Deep) Reinforcement Learning
- Generative Adversarial Networks (GANs)
- Graph Neural Networks (GNNs)
- Adversarial Learning
- Transfer Learning
- AutoML – Automated Machine Learning
How to store and **re-use generated knowledge** between different but related problems

Also applied when **fine-tuning a pre-trained model**
Other Relevant Learning Topics

- NLP – Recurrent Neural Networks (RNNs) and More (Attention, Transformers)
- (Deep) Reinforcement Learning
- Generative Adversarial Networks (GANs)
- Graph Neural Networks (GNNs)
- Adversarial Learning
- Transfer Learning
- AutoML – Automated Machine Learning
AutoML – Automated Machine Learning

- In general, **performance of a ML algorithm** heavily depends on the **many hyperparameters** involved in the end-to-end analysis pipeline.

- Optimization algorithm, learning rates, momentum, batch normalization, batch sizes, dropout rates, weight decay, data augmentation, etc..

- **Easily 30 to 40 design decisions!**
AutoML – Automated Machine Learning

- Common approach: some sort of search approach (optimization) to find best hyperparameters (e.g., grid-search)

- **AutoML – *AI to build AI***: why not relying on ML to find best end-to-end ML pipeline configuration?

### Current deep learning practice

1. Expert chooses architecture & hyperparameters
2. Deep learning “end-to-end”

### AutoML: true end-to-end learning

1. Meta-level learning & optimization
2. Learning box
AutoML – Automated Machine Learning

- **AutoML** is an emerging AI/ML discipline which attempts to **automate the end-to-end** process of applying **machine learning** to **real-world problems** *(concept drifts)*

- **Traditional ML pipeline:**
  - clean and pre-process data
  - engineer features
  - select a model family
  - set the hyperparameters
  - construct ensemble of models
  - ...
  - **Relies in domain knowledge!**

- **Auto-ML**
  - Hyperparameter optimization and model selection
  - **Full pipeline optimization**
  - Deep neural network architecture search
  - Libraries such as auto-WEKA, auto-sklearn, auto-Keras, etc.

- **AutoML @NeurIPS Challenge**
AutoML – Neural Architecture Search (NAS)

- NAS is a subset of AutoML methods
- NAS is an approach to *automatically build network architectures*
- Has been used to design networks that are on par or outperform hand-designed architectures
- Almost 2% of ICLR 2020 submitted papers using NAS
Q&A...

AIT Austrian Institute of Technology @Vienna

https://bigdama.ait.ac.at/

pedro.casas@ait.ac.at
http://pcasas.info