Technical Keynote Lecture:

Multi-Task Deep Learning
Prof. Giuseppe Di Fatta

Department of Computer Science
University of Reading
G.DiFatta@reading.ac.uk
Acknowledgments

- Prof. Giuseppe Nicosia, University of Catania and University of Cambridge
- Deyan Dyankov, MSc graduate, University of Reading, and UBS, London
- Salvatore Danilo Riccio, PhD student, Queen Mary University of London
- Giorgio Jansen, University of Cambridge
Overview

- Stein’s Paradox in Statistics
- Overview of Transfer Learning
  - Approaches and challenges
- Multi-Task Deep Learning
- Multi-Task/Multi-Objective Optimisation in Deep Neuroevolution
- Conclusions
Stein’s Paradox in Statistics

• The best estimation of unobserved quantities is their observed averages.
  – The mean minimises the squared error over the samples.
  – The sample mean may differ from the population mean, but cannot do any better from a sample set of a single variable.

• There is a better estimator than sample average when estimating **multiple independent** Gaussian random variables:
  – Arguably the first form of multi-task learning

*Charles Stein, “Inadmissibility of the usual estimator for the mean of a multivariate normal distribution”, TR, Stanford University, 1956*
Stein’s Paradox: Example

- Baseball players’ performance
  - **Hit**: the batter strikes the ball and safely reaches or passes the first base
  - **Unobserved quantity**: true batting performance in a season
  - **Observed quantity**: batting average in the first N bat times

\[
\text{Batting average} = \frac{\text{hits}}{\text{bat times}}
\]

- **Paradox**: the best estimator for each player’s expected performance is **NOT** their individual observed batting average.
1970 Major-League Baseball Players

James-Stein Theorem (1961): for $k>3$ the JS estimator dominates the sample average.

Squared Errors

Batting Averages

The Shrinking Factor

• The shrinking factor in James-Stein estimator is a form of **multi-task regularisation** for averages.

• For a number k (k>3) of independent variables (players):

\[
\mu_i^{js} = \bar{\mu} + c(\mu_i - \bar{\mu})
\]

\[
c = 1 - \frac{(k-3)\sigma^2}{\sum(\mu_i - \bar{\mu})^2}
\]

where:
- \(\mu_i^{js}\): JS estimator
- \(\bar{\mu}\): grand average
- \(\mu_i\): single average
- \(k, \sigma\): known and fixed

*Source: Bradley Efron and Carl Morris, “Stein’s Paradox in Statistics”, 1977*
Single-Task Learning

• many binary classifiers
Multi-Task Learning (MTL)

- a single shared model
Multi-Task Learning (MTL)

- In MTL, tasks are trained in parallel using a shared representation.
- An inductive transfer mechanism improves generalization performance by leveraging domain information from related tasks.
- The training data for many tasks work as an inductive bias: learning all tasks concurrently helps each task to be learned better and reduces the risk of overfitting on any of them.

“MTL is a collection of ideas, techniques, and algorithms, not one algorithm.”

MTL Example: Multinominal Classification

- Different class labels as multiple tasks
- Object recognition from images: cat, dog, rabbit, etc.
- CNN layers: from generic features to specific ones
- Dense layers for classification
Example: Medis Pneumonia Data

- Predict mortality risk for hospitalisation

Symptoms, risk factors
- Age
- Asthmatic
- Chest pain
- Wheezing
- Stridor
- Hearth murmur
- Diabetic
- etc.

Laboratory tests
- Hematocrit
- White blood cell count
- Potassium
- etc.

Intensive care
- Mortality risk
STL on Medis Pneumonia Data

- Predict mortality risk for hospitalisation [Caruana, 1997]

```
available initial inputs
Input layer  Hidden layer  Output layer
Age  Sex  Asthmatic  Chest Pain  Wheezing  Stridor  Heart Mumur  Diabetic

Mortality Rank

```
MTL on Medis Pneumonia Data

- Predict mortality risk for hospitalisation [Caruana, 1997]

MTL Internal Mechanisms

• **Data amplification for learning shared features**
  – Limited data and noisy data for some task: multiple tasks contribute to increase the overall sample size.
  – Aggregated data help by averaging the noise in individual datasets.

• **Eavesdropping**
  – When noise in one task T’ is too high to learn a particular feature, other tasks help generating the feature that may be useful also in T’.

• **Representation bias**
  – MTL drives the stochastic search towards a better (more general) solution avoiding local minima on individual tasks.

• **Unsupervised discovery of task relatedness**
  – Task relatedness can be discovered by the learning process, effectively performing an implicit unsupervised process within the main supervised process.
  – Backpropagation using shared layers can exploit the way multiple tasks are related without being given explicit information about task relatedness.
Transfer Learning Approaches

• **Sequential Transfer Learning**
  – Tasks are learned sequentially
  – The final model works well for the target task

• **Multi-Task Transfer Learning**
  – A single model for 2 or more target tasks is learned concurrently
  – The final model works well for all the tasks

• **Multiform Transfer Learning**
  – It refers to a form of multi-task transfer learning in which multiple formulations of a single task are learned: task engineering.
  – Transforming a single-objective optimization problem into a multi-objective optimization problem can help to remove local optima.

A. Gupta, Y. Ong and L. Feng, *Insights on Transfer Optimization: Because Experience is the Best Teacher,* IEEE Transactions on Emerging Topics in Computational Intelligence, 2018.
Sequential Transfer Learning: Feature Extraction

- Source task A

- Target task B

Frozen: feature extractor

Features

ML algorithm training

Backprop
Sequential Transfer Learning: Fine Tuning

- Source task A

- Target task B

Backprop

Frozen
Pre-Trained Models: Examples

- **Pre-trained models for Computer Vision:**
  - VGG-16, VGG-19, Inception V3, Xception, ResNet-50, etc.

- **Word embedding pre-trained models for NLP tasks:**
  - Word2Vec, GloVe, FastText

- **Other models for NLP that can be used for transfer learning:**
  - Universal Sentence Encoder (Google)
  - Bidirectional Encoder Representations from Transformers (BERT, Google)
MTL Soft Regularisation

Task 1-specific layers

Task 2-specific layers

soft regularisation constraint
MTL Hard Regularisation

Task-specific layers

Shared representation layers

Backprop

A

B

hard regularisation constraint

$L_A$

$L_B$

Aggregated Loss
MTL Challenges

- **Feature transferability** and **task relatedness**
  - Which features should be shared/transferred?
    - Issues: *negative-transfer* in feature layers and *under-transfer* in classifier layers
  
  - How to measure task similarity?
  
  - How to embed known task relations?
    - What if prior knowledge on task relations is not available? Can MTL still be employed to learn **unknown or unexpected** task relations?
Feature Transferability

Transferability is negatively affected by two issues:

– optimization difficulties in splitting networks between co-adapted neurons
– the specialization of higher layer neurons to their original task at the expense of performance on the target task

Experiments [Yosinski, 2014] on a network trained on ImageNet: features transferred from the bottom, middle, or top of the network

– The transferability of features decreases when the base task and target task are less similar.
– Nevertheless, transferring features even from relatively dissimilar tasks can be better than using random features.
– Transferring features from almost any number of layers can produce a boost to generalization that lingers even after fine-tuning.

Task Relatedness

• MTL is expected to work if tasks are related and share some common features.
  – How can we determine if tasks are sufficiently related?
  – In multi-label classification this may not always be the case.

• Task similarity (metrics) to quantify the relatedness of tasks
  – Assuming explicit similarity matrix (e.g., S. Feldman, M. R. Gupta, and B. A. Frigyik, “Multi-Task Averaging”, NIPS 2012) to describes the relatedness of any pair of tasks.

• Explicit prior structure of task groupings
  – Explicit hierarchical task relatedness, with prior knowledge or with some explicit structure in learning relations

• No prior knowledge of task relatedness
  – Implicit hierarchical task relatedness, no prior knowledge of learning relations
  – Learning the task relations

<table>
<thead>
<tr>
<th>sequential</th>
<th>hierarchical</th>
<th>clique</th>
<th>star</th>
<th>unknown connected graph</th>
<th>connected components</th>
</tr>
</thead>
</table>

Giuseppe Di Fatta
Tasks Structure

• How to incorporate the tasks structure in the learning problem?
  – A custom model and/or a custom Loss function. In general, a combination of
    the loss functions from multiple heads. E.g., Multilinear Relationship Networks:

\[
\min_{f_t} \sum_{n=1}^{N_t} J \left( f_t \left( x^t_{n} \right), y^t_{n} \right)
\]

- A linear combination \( \sum_{i} \alpha_i \mathcal{L}_i \) with weights \( \alpha_i \): more hyperparameters!

- A meta-learning problem: a general regularization framework to learn multiple
  tasks as well as their structure.
  • E.g., “Convex Learning of Multiple Tasks and their Structure” (Ciliberto 2015)
Top-Down Layer-wise Model Widening

- An automated procedure to progressively learn the grouping of tasks (clustering) in the model structure
  - each branch is associated with a sub-set of tasks.

Combining Loss Functions

Image style transfer task

- Training from a distribution of loss functions
- Testing with a conditional coefficient

Source: https://ai.googleblog.com
- Alexey Dosovitskiy, Josip Djolonga, You only train once: loss-conditional training of deep networks, ICLR 2020
- Mohammad Babaeizadeh and Golnaz Ghiasi, Adjustable Real-Time Style Transfer, ICLR 2020
Combining Loss Functions

Adjustable style transfer

- All stylizations are generated with a single network by varying the conditioning values.

Source: https://ai.googleblog.com
- Alexey Dosovitskiy, Josip Djolonga, You only train once: loss-conditional training of deep networks, ICLR 2020
- Mohammad Babaeizadeh and Golnaz Ghiasi, Adjustable Real-Time Style Transfer, ICLR 2020
Multi-Task/Multi-Objective Optimisation

- MTL is intrinsically a multi-objective problem
  - different task objectives may be consistent or may conflict with each other
    - minimisation of a single objective function as linear combination
    - trade-off of many objective functions

- Optimisation approaches
  - Using prior knowledge, then **Bayesian optimisation** methods can be applied:
    - knowledge transfer/sharing for a faster automatic hyperparameter optimization in ML
    - can improve the efficiency of training and the generalization capability of models
  - No prior knowledge: **Evolutionary approaches** for a unified search space for all tasks.
    - modes of knowledge transfer include shared genetic makeup, direct genetic crossover, other methods w/o direct solution crossover.
  - Find a trade-off using **Pareto Multi-Objective Optimisation**
    - Pareto optimal solutions (Pareto front)
Pareto Multi-Task Deep Learning

➢ Population-based algorithms, such as Neuroevolution, can be naturally extended to multi-task learning.
  – Use ES to concurrently optimise many tasks and many objectives with a single DNN model.

➢ Multi-Task Multi-Objective Deep Neuroevolution with a Pareto optimisation approach
  – Tasks selected from related Atari 2600 games
  – Prior knowledge used to define multiple utility functions
  – Analysis of the underlying training dynamics with standard techniques and with the Hypervolume indicator and the Kullback-Leibler divergence

➢ Results: a single model trained on multiple games outperforms models trained on individual games.

Deep Neuroevolution

- Approximated loss gradient for backpropagation
  - It can compete against gradient-descent deep learning algorithms in terms of performance in difficult reinforcement learning problems

- Offspring are generated from a gaussian distribution centered at the best parent from the previous generation.
  - **mirrored sampling**: evaluating the mirrored offspring allows to estimate the gradient without differentiation.

\[
\theta_{P(k+1)} = g(\theta_{P(k)}, \nabla \theta f(\theta(k))) , \theta(k) \in \Theta_k
\]

- $\theta_{P(k+1)}$: parent model parameters at generation $k+1$
- $P$: parent
- $f$: distribution for the offspring sampling
- $g$: optimiser
- $\nabla \theta$: estimated gradient
Multi-Task Multi-Objective Evolutionary Strategy (MTMO-ES)

- Assuming a general association between a task (utility, goal) and multiple objectives (features): m tasks associated to n objectives.
- We can reformulate the Deep Neuroevolution approach to the multi-task multi-objective case with the MTMO-ES gradient:
  - a weighted sum of the gradients associated to the objectives.

Matrix (m tasks x n objectives):
\( \delta_{ij} \): association matrix (binary values) between tasks (i) and features (j)

\[
\begin{array}{ccccccc}
\text{task} & f_1 & f_2 & f_3 & \ldots & f_n \\
t_1 & 1 & 1 & 0 & \ldots & 0 \\
t_2 & 0 & 0 & 1 & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
t_m & 0 & 0 & 0 & \ldots & 1 \\
\sum & 1 & 1 & 1 & 1 & 1 & 1
\end{array}
\]

\[
\theta_P(k + 1) = g(\theta_P(k), \sum_{i=1}^{n}\alpha_i \nabla_{\theta}f_i(\theta(k))) , \theta(k) \in \Theta_k
\]

\[
\sum_{i=1}^{n}|\alpha_i| = 1 , \alpha_i \in \mathbb{R} \forall i
\]

\[
\sum_{i=1}^{n}|\alpha_i| \delta_{ij} = \frac{1}{t} , \forall j \in \{1, \ldots, t\}
\]
The ANN (Deep Q-Network) learns a “policy” to play the game from video-only input. It has learned to avoid incoming fire and to move to shoot the lower-down aliens first.

This video was created from the policy being trained with 5.95M frames, which takes a few days using a high-end server with a powerful GFX card.

Matthew Uzzell
MSc Advanced Computer Science, 2017-18
Department of Computer Science
University of Reading
Experimental Analysis

- DL architecture:
  - input layer, three convolutional layers, a fully connected layer, a fully connected output layer (18 actions)
- Two games with similar dynamics: River Raid and Zaxxon
Training: Elite Scores

- Multi-task single-objective ES trained on both tasks and evaluated on each task.
  - Mean score: each model is evaluated over 200 episodes.
  - Elite mean scores, with minimum and maximum score per iteration.
  - The dashed lines are the results obtained by single task ES (*).

Training: Offspring Scores

- Multi-task single-objective ES trained on both tasks and evaluated on each task.
  - Mean score: each model is evaluated over 200 episodes.
  - 5000 offspring mean scores, with minimum and maximum score per iteration.
  - The dashed lines are the results obtained by single task ES (*).

The **Pareto front** is the set of solutions $\theta'$ not dominated by any other solution:

$$P(\Theta) = \{ \theta' \in \Theta \mid \nexists \theta \in \Theta : \theta \leq \theta' \}$$
Pareto Front Hypervolume

- The Pareto front obtained with multi-task ES covers a larger area than the two single-task single-objective networks.
  - The new algorithm is finding strategies able to master both tasks at the same time.
Conclusions

- Sequential transfer learning more frequently used to transfer features from few established tasks with lots of data to many other applications with limited data.
  - becoming a key methodology for mainstream adoption of DL in industry with pre-trained models in computer vision and speech processing.

- The main idea of Multi-Task Learning (MTL) is exploiting the relations/structure among different tasks: less frequently adopted.
  - inherent difficulty to identify and group "related" problems systematically and our tendency to think in terms of “weak” AI

- MTL: a step forward in learning paradigms
  - single-task machine learning exploits complex relations in the data
  - MTL also exploits complex relations in the tasks

- Machine Learning has made huge progress in solving isolated problems: MTL is now inspiring research on more general deep learning architectures.