# Project Plan: Reinforcement Learning in Latent Space

We, the authors

October 8, 2018

## 1 Context and Motivation

In **Reinforcement Learning** (RL), machine learning problems are modelled as a sequence of actions taken by an agent in some environment to maximize a total reward. Instead of learning from a dataset, the agent builds knowledge about the environment and optimal actions by exploring the effect of its behaviour on received rewards. Since such interaction can be cost-intensive in real-world (or physical) applications, it is desirable to pre-train agents on a simulated task and afterwards generalize the obtained knowledge to the real task.

For this reason, recent research on reinforcement learning involves a lot of work on **transfer learning** (TL). In TL, an agent learns to perform one task and uses its knowledge in a before unseen task to perform to a reasonable level with minimal additional training. In its more extreme forms, TL is known as one- or zero-shot-learning, when only a single training step is allowed in the target task - or none at all (Goodfellow, Bengio, and Courville, 2016). In RL, one of the key challenges of applying TL is to align states and actions between tasks. While some work tackles this issue with hand-crafted solutions (e.g. Taylor and Stone, 2007), it is desirable to develop methods to automatically map tasks.

**Deep Learning** (DL) has been successfully applied to a variety of problems in machine learning research and got increasing attention over the last two decades (Goodfellow, Bengio, and Courville, 2016). While DL is applicable to classical problems such as regression and classification - and may henceforth be used as a policy learner in RL as well - it is particularly useful for learning representations in a latent space. For instance, convolutional neural networks can be used to break down visual input into features modelling higher-level information. Sequences of variable length, such as natural language, can be embedded using recurrent neural networks (Goldberg and Hirst, 2017). Another architecture for representation learning are so called *autoencoders* (Hinton and Salakhutdinov, 2006b). These neural networks learn to first reduce the dimensionality of their inputs and then reconstruct the original sample from this low dimensional representation. The low-dimensional representation in the middle layer often proofs useful as a representation for different tasks as it encodes the relevant information of the input space into a latent space.

---

Describes the problem/task/goal/idea in a general context

- Domain
- Relevance
- State of the Art
- Open issues

- Objectives
- Research Questions

## 2 Concepts and approach

**Deep learning (DL)** is a specific subfield of machine learning that aim to learn representations of the world from sets of high level features built by the composition of lower level ones. The main aspect of deep learning is that these set of features, called layers, are learned from raw data through general-purpose procedures and without the intervention of humans.

The algorithms that allow this kind of learning are the **artificial neural networks** which are a mathematical function that maps a set of inputs values to outputs values (Goodfellow, Bengio, and Courville, 2016). Such function is a composition of many simpler functions that provides several levels of abstractions.

**Convolutional neural networks (CNNs)** are a category of artificial neural networks designed to process "grid-like topology data" (Goodfellow, Bengio, and Courville, 2016) like images or sounds signals. Its architecture, that take advantage of the topology of the input, arrange the neurons in three dimensions: width, height and depth.

The name of this kind of neural networks comes from the operation in which they are based, the convolution, which is a special kind of linear operation. CNNs architecture is formed by four main layers:

- Convolutional layer.
- ReLu layer.
- Pooling layer.
- Classification layer.

The convolution layer is the main part of the CNNs, being in charge of detecting the features in the data by the use of filters. To improve the speed of the training, convolution layers are followed by the rectified linear units (ReLu) layer, that acts as a rectifier function.

The functionality of the pooling layer is reduce the spatial size of the representation with the objective of decrease the parameters in the network and the computation required. Finally, the last part of the network is the fully connected layer, which performs the classification task.

An **autoencoder** is neural networks trained to copy its input to the output and which is formed by two parts: the encoder, that converts the input to a dense and smaller representation and the decoder, that rebuild the input from the representation. The main application for autoencoders is the dimensionality reduction (Hinton and Salakhutdinov, 2006a), where the encoder learns to preserve the meaningful attributes of the input and generates a lower dimensional representation that is saved in the **latent space**.

One of the limitations of generic autoencoders is that the generated latent space is based in features of the data and may not be continuous. This lack of continuity works well for the replication of the input but produce unrealistic outputs in generative tasks or with inputs that has not been previously observed.

In order to create a continuous latent space, **Variational autoencoders (VAEs)** use probability distributions to describe each attribute, allowing interpolation and random sampling in the data.

Moreover, VAEs are trained with **gradient based methods** which gives a better control over the latent space representation (Goodfellow, Bengio, and Courville, 2016) and generates representations with disentangled factors (Higgins et al., 2016).

**Latent Space (LS)** is the space in which information generated by the encoder lies. It contains the compressed representation of the input that can be used by the decoder to regenerate the data.

---

How will you be tackling/solving this?
Concepts:

- Existing Technologies
- Methods
- Terminology

Approach:

- What will you be doing with these technologies?
- How will you apply them?

# 3 Deliverables

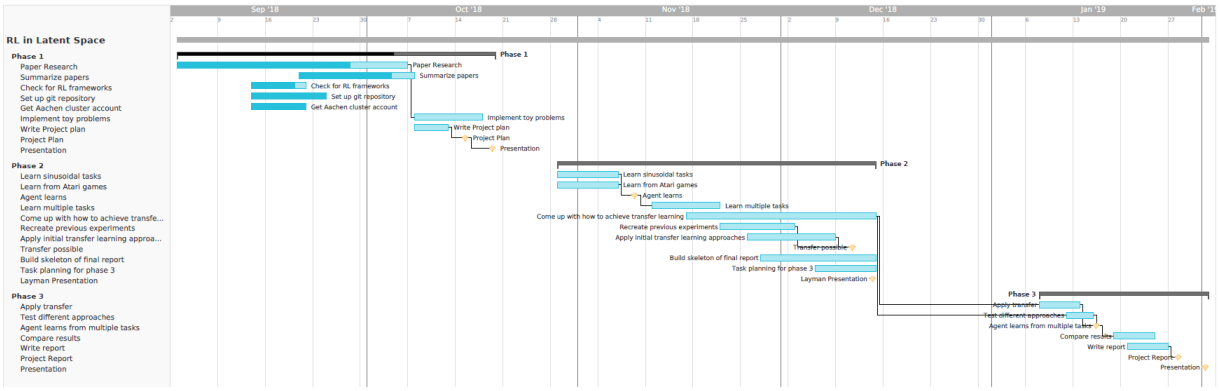
What we want to deliver is a method to transfer learning between different tasks. For this we will develop a framework for training and transferring knowledge from one task to another. The tasks need to have some similarities, otherwise you would not get a benefit from using previous knowledge. An interesting application are games in which success is evaluated using the score the agent achieved. We will show that transfer learning helps the agent to adapt to a new environment quicker than using no prior knowledge. Furthermore, we want to give an insight into how transfer learning can be achieved and which important aspects need to be regarded to acquire a positive effect.

# 4 Time Management

In order to tackle the project efficiently we dedicate a lot of time of the first phase into researching previous papers that already looked into transferring learning between different tasks. Furthermore, we will look for useful reinforcement learning frameworks which we can use to test our method. To verify that we sufficiently know these frameworks we will have to implement some toy problems which should be done by the end of phase 1. In phase 2 we will split our group up into two groups to focus on different tasks that the agent will be trained on. One of these approaches is an easier implementation of learning as a fall-back if the other one does not work out. The main focus in this phase lies on coming up with a method to transfer learning between different tasks. Towards the end of this phase we plan to recreate existing transfer learning experiments and maybe even implementing our own first attempts. Furthermore, we will have a skeleton for our final report that we have to hand in in phase 3. Phase 3 will then focus on applying and implementing our ideas to transfer learning. After achieving this, we will compare our results with already existing ones and start writing our final report.

### 4.1 Gantt Chart

To illustrate, we created a Gantt chart to show our task planning.




## 5 Risk Analysis

We distinguish between technical and organizational risks, where the former involve those related to the approach chosen to tackle the research problem, whereas the latter refer to those arising from the organization of work of the project team and related parties. In addition to the descriptions and contingency plans, justifications for the costs and likelihoods of the risks are provided.

### 5.1 Technical Risks

| ID | Risk Description | Contingency Plan | Cost | Likelihood |
|---|---|---|---|---|
| 1 | The team fails to build the RNN- or CNN-based model from TIMIT on time. | Prioritize this task over other tasks to speed up the implementation. For others tasks of which this task is a prerequisite, reallocate the assigned manpower to items with no dependency on this task. | +++ | + |

### 5.2 Organizational Risks

| ID | Risk Description | Contingency Plan | Cost | Likelihood |
|---|---|---|---|---|
| 1 | Testing data from children's speech cannot be collected before the methods need to be tested. | Re-launch data collection with more accessible test subjects, such as university students of younger age. | +++ | + |

### 5.3 Cost and Likelihood Explanation

| ID | Cost Explanation | Likelihood Explanation |
|---|---|---|
| 1 | +: Factors causing the long-term absence of a teammate are rare. | ++: The absence would cause a 20% drop in manpower. |

# References


Goldberg, Yoav and Graeme Hirst (2017). *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers. ISBN: 1627052984, 9781627052986.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Higgins, I. et al. (June 2016). "Early Visual Concept Learning with Unsupervised Deep Learning". In: *ArXiv e-prints*. arXiv: 1606.05579 [stat.ML].

Hinton, G. E. and R. R. Salakhutdinov (2006a). "Reducing the Dimensionality of Data with Neural Networks". In: *Science* 313.5786, pp. 504–507. ISSN: 0036-8075. DOI: 10.1126/science.1127647. eprint: http://science.sciencemag.org/content/313/5786/504.full.pdf. URL: http://science.sciencemag.org/content/313/5786/504.

Hinton, Geoffrey E and Ruslan R Salakhutdinov (2006b). "Reducing the dimensionality of data with neural networks". In: *science* 313.5786, pp. 504–507.

Taylor, Matthew E. and Peter Stone (2007). "Cross-domain Transfer for Reinforcement Learning". In: *Proceedings of the 24th International Conference on Machine Learning*. ICML '07. Corvalis, Oregon, USA: ACM, pp. 879–886. ISBN: 978-1-59593-793-3. DOI: 10.1145/1273496.1273607. URL: http://doi.acm.org/10.1145/1273496.1273607.