Knowledge-Defined Networking:
Towards Self-Driving Networks

Albert Cabellos (UPC/BarcelonaTech, Spain)
albert.cabellos@gmail.com

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

2nd IFIP/IEEE International Workshop on Analytics for Network and Service Management
AnNet 2017
Lisbon, May 2017

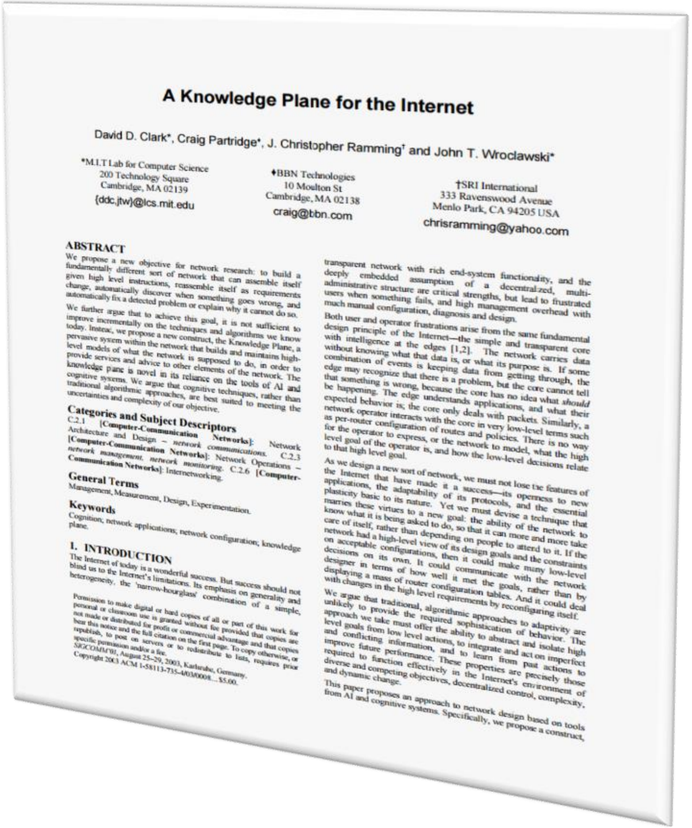
Contextualization & Motivation
Applying Machine Learning to Networks

A Knowledge Plane for the Internet

D. Clark (MIT) “A Knowledge Plane for the Internet”, 2003

“*we propose a new construct, the Knowledge Plane, a pervasive system within the network that builds and maintains high-level models of what the network is supposed to do*”

“*The knowledge plane is novel in its reliance on the tools of AI and cognitive systems*.”


A Knowledge Plane for the Internet

David D. Clark*, Craig Partridge*, J. Christopher Ramming† and John T. Wroclawski*

*M.I.T Lab for Computer Science, 200 Technology Square, Cambridge, MA 02139, {ddc,jtw}@lcs.mit.edu

BBN Technologies, 10 Moulton St, Cambridge, MA 02138, craig@bbn.com

†SRI International, 333 Ravenswood Avenue, Menlo Park, CA 94205 USA, chrisramming@yahoo.com

ABSTRACT

Categories and Subject Descriptors

General Terms

Keywords

1. INTRODUCTION

Clark, David D., et al. "A knowledge plane for the internet." *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2003.

Why we are not there?

- Traditionally networks have been **distributed** systems
  - Partial view and control
- Beyond programmability, SDN provides **centralization**:
  - **Full control** over the network
- Network Analytics/Telemetry provides **full view** of the network
- SDN and NA are **enabling technologies** for Machine Learning techniques applied to networks

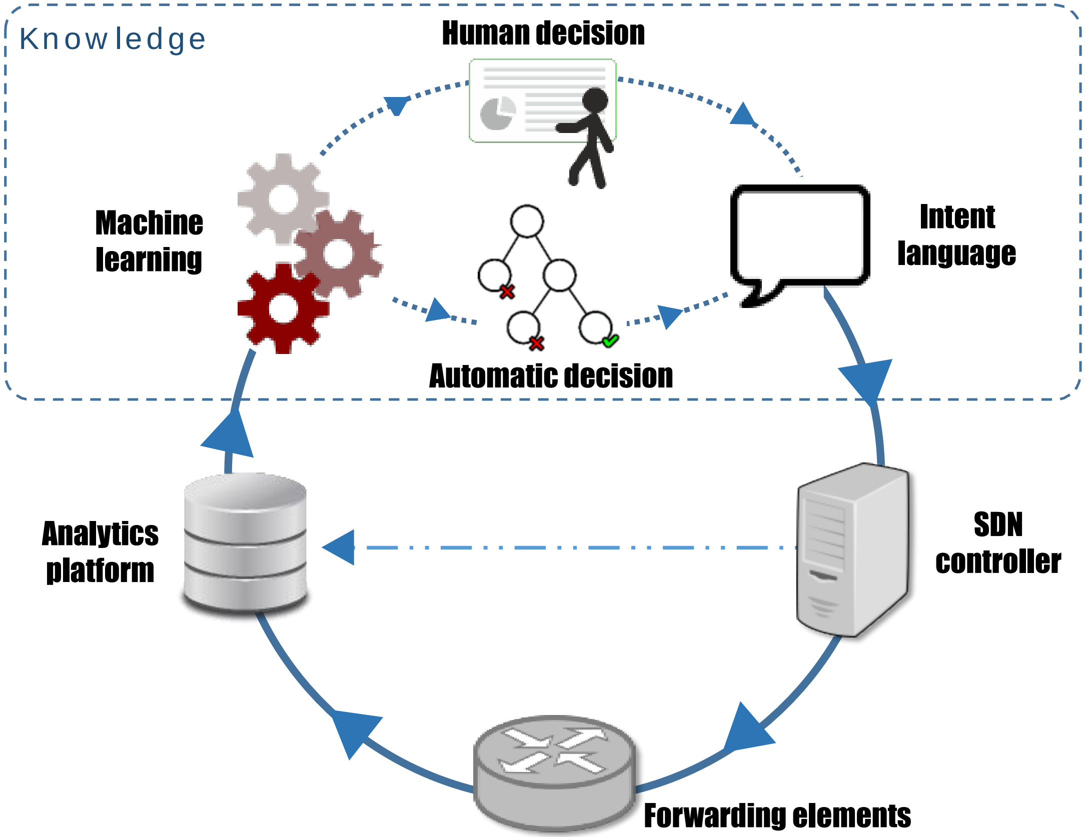
Knowledge-Defined Networking

- Apply ML techniques to Networking:
  - Control (fast dynamics)
    - E.g, routing, resource allocation (NFV/SFC), PCE, optimization, congestion detection
  - Management (slow dynamics)
    - E.g., network planning, load estimation
  - Recommendation mechanisms
  - Out of the scope: Traffic Analysis, Anomaly Detection, Root-Cause Analysis
- Towards **self-driving networks**

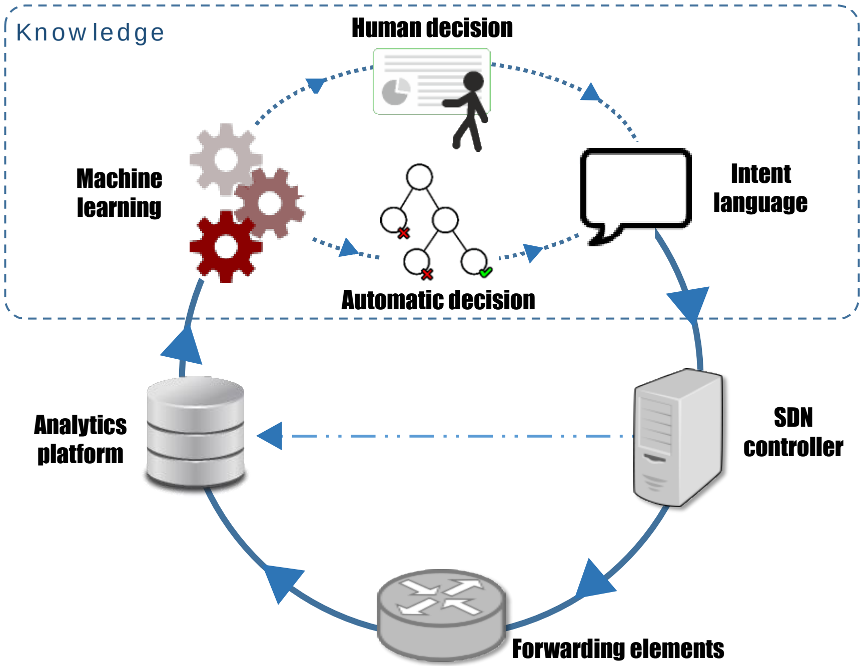
Evolution in other fields

Hardware → Software → Artificial Intelligence

Knowledge-Defined Networking Paradigm



Goals of KDN



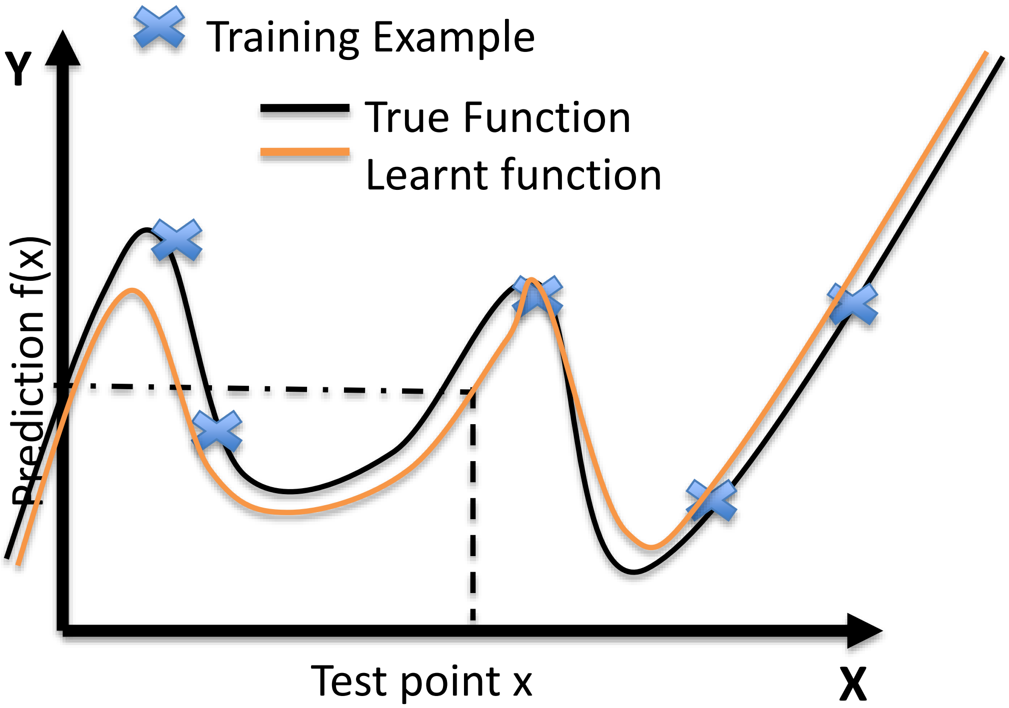
- Recommendation
- Optimization
  - Hidden Information
  - Complex systems
- Estimation/Validation
  - Performance/Cost
- **Automatization**

Scope of this talk

- How you can build a self-driving network?
- Show some preliminary results
- Outcomes:
  - Which are the main challenges?
  - New research directions
  - **Have a better understanding of what are the consequences of using ML to control the network**

A brief intro to Machine Learning

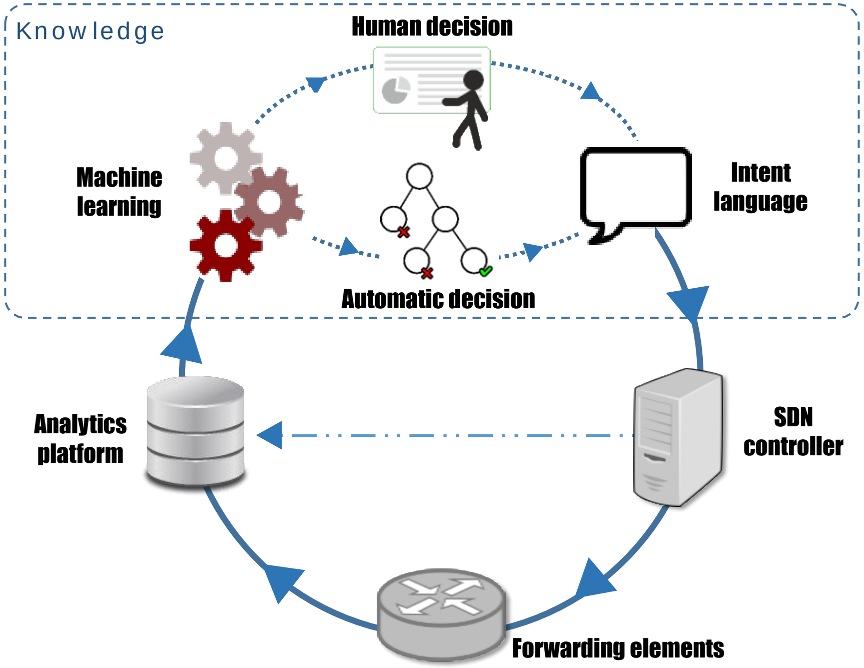
ML fitting the (true) function

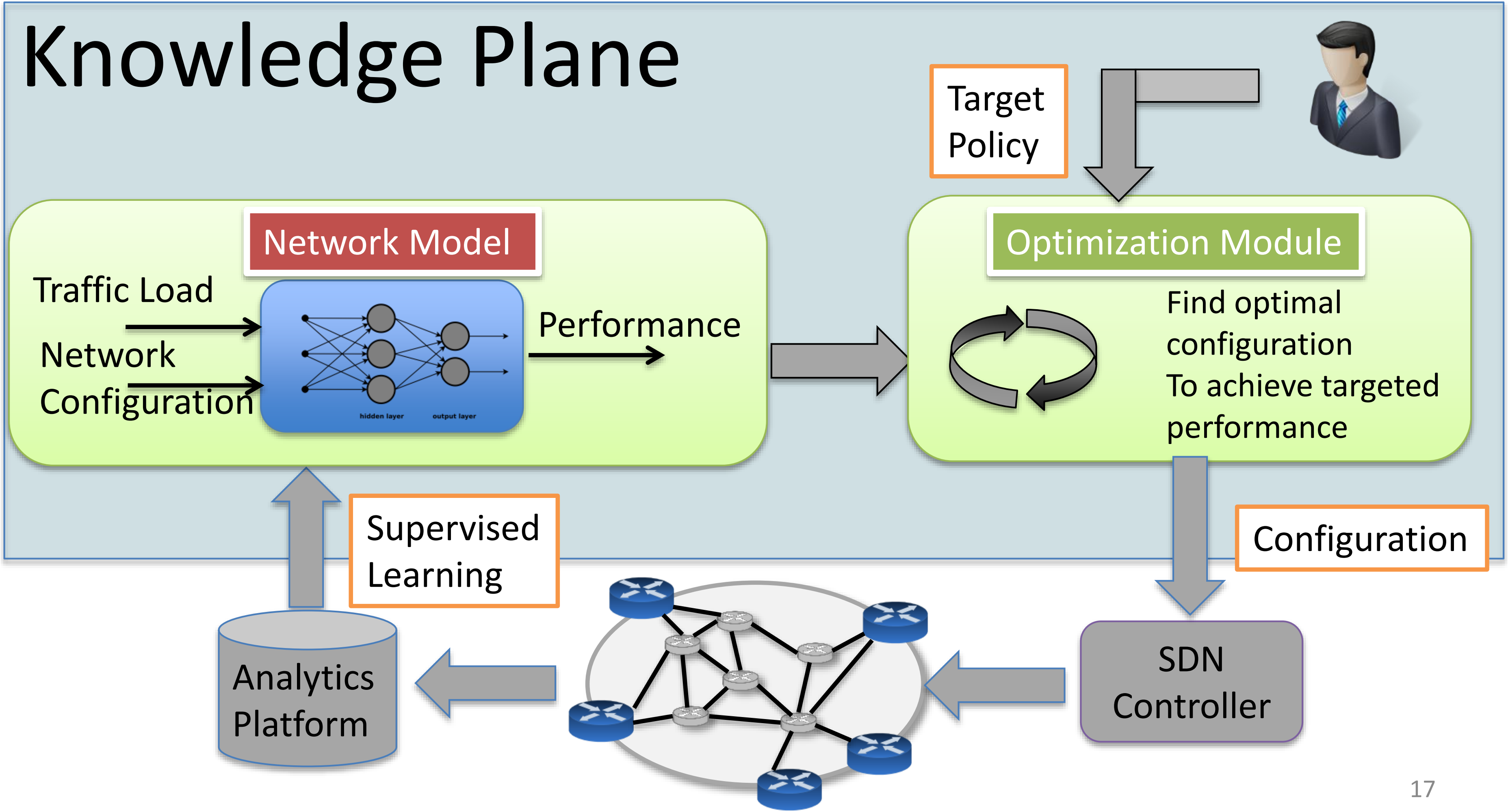


- With enough data ML will fit the true function
- ML interpolates and extrapolates
- Can predict unseen scenarios

Building a self-driving network

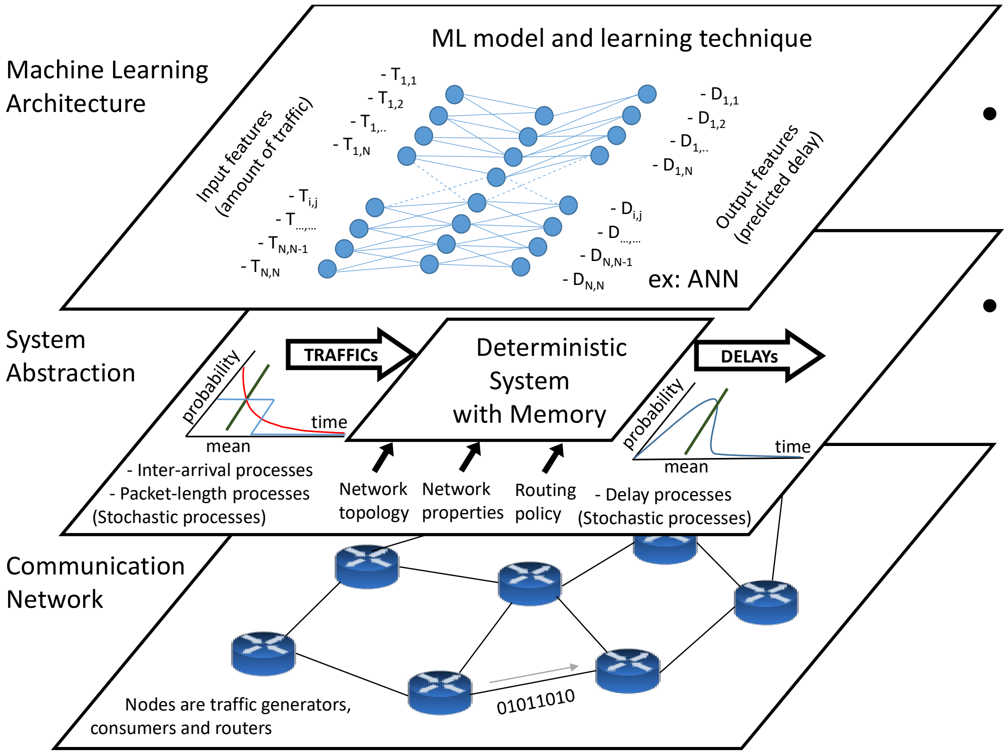
Building a Self-Driving Network




Knowledge Plane
Target Policy
Network Model
Traffic Load
Network Configuration
hidden layer
output layer
Performance
Optimization Module
Find optimal configuration
To achieve targeted performance
Supervised Learning
Configuration
Analytics Platform
SDN Controller

Network Modeling based on Machine Learning

Network Modeling based on Machine Learning

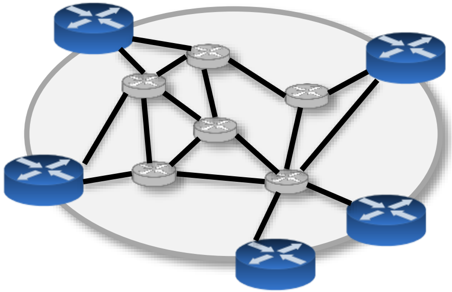


- It is feasible to model the network as a black-box using Machine Learning techniques?
- Can we build a model using Machine Learning techniques that given:
  - Configuration of the network
  - Traffic Load

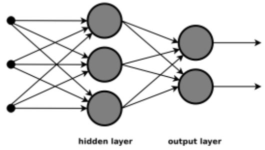
  Is able to accurately estimate the delay?

Methodology

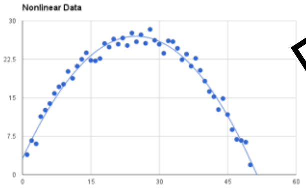
Training Set



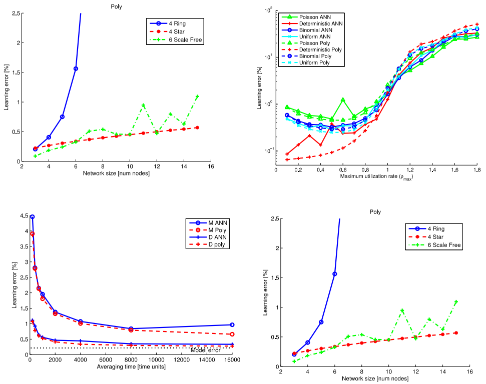
Supervised Learning



- SVM
- ANN
- Polynomial R.



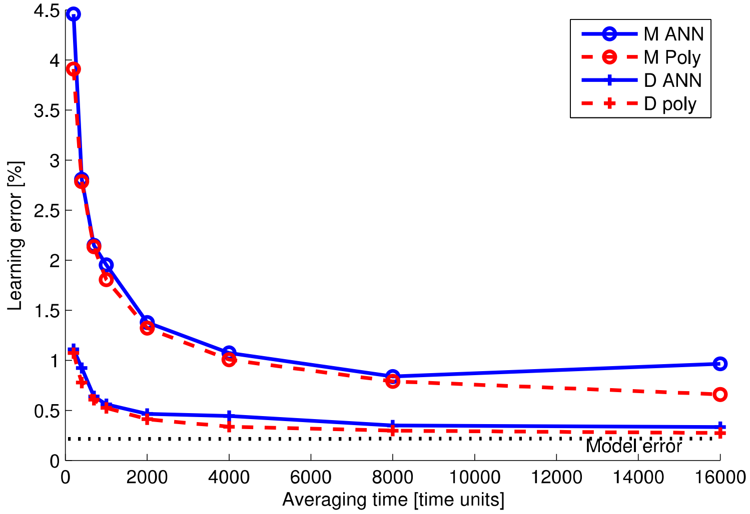
- Topology
- Network Size
- Traffic Characteristics
- Saturation
- And the resulting delay

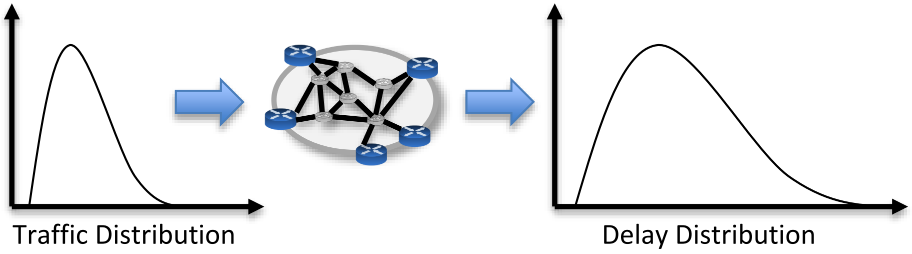


Evaluate the accuracy

Which is the relation between accuracy and the fundamental network properties?

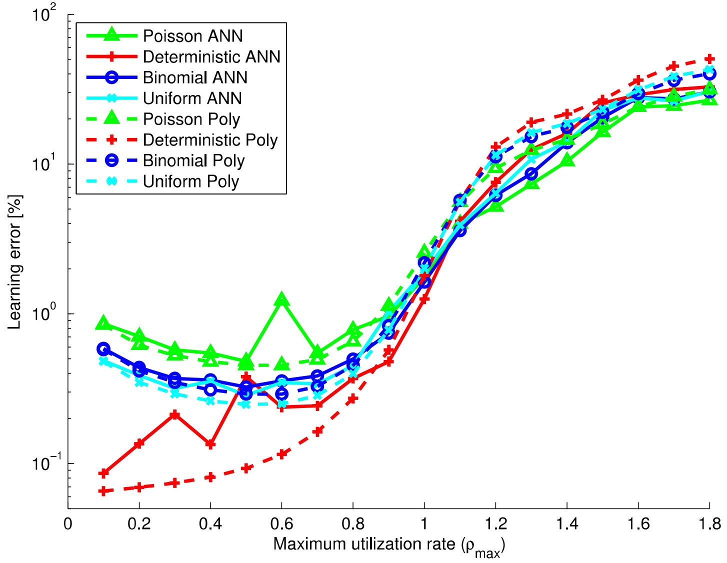
Results: Variance of the Data-Set



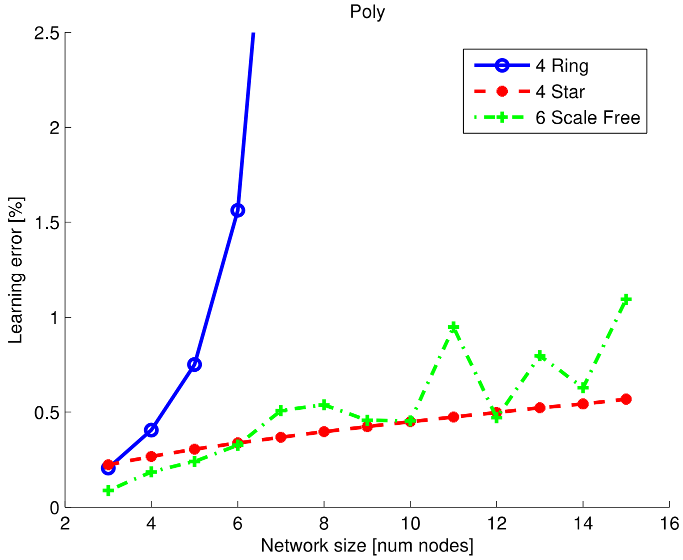


- ML models the network with an error <1%
- Some network characteristics increase the variance of the delay
- Lager variance requires more samples to accurately estimate the end-to-end latency

Results: Variance of the Data-Set

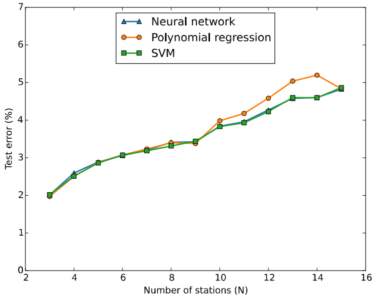


Traffic Intensity

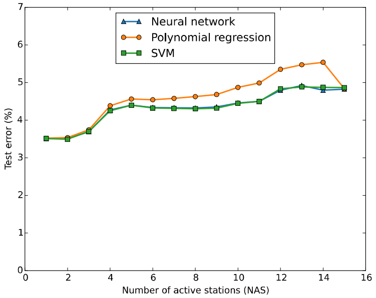


Network Topology

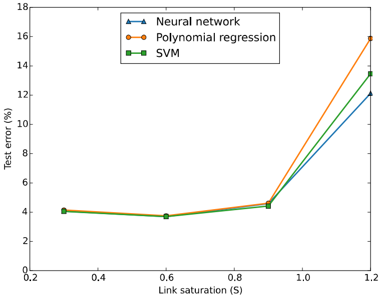
Results with appropiate averaging times



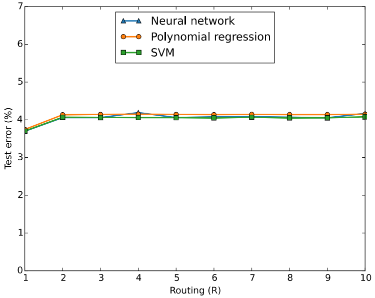
(a) Error as a function of the number of stations



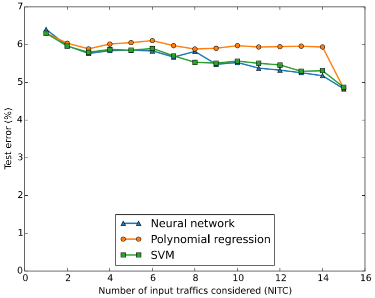
(b) Error as a function of the number of active (transmitting) stations



(c) Error as a function of the saturation level (average ρ)



(d) Error as a function of the routing policy



(e) Error as a function of stations considered when training the models

Discussion

- Machine Learning is a **third-pillar** in Network Modeling
  - Analytical Techniques (e.g., Markov Chains)
    - Do not work well in complex scenarios
  - Computational Models (e.g., Simulation)
    - High cost in terms of CPU
    - Simulating complex networks requires costly development
  - Neural Networks

Machine Learning for Computer Network modeling

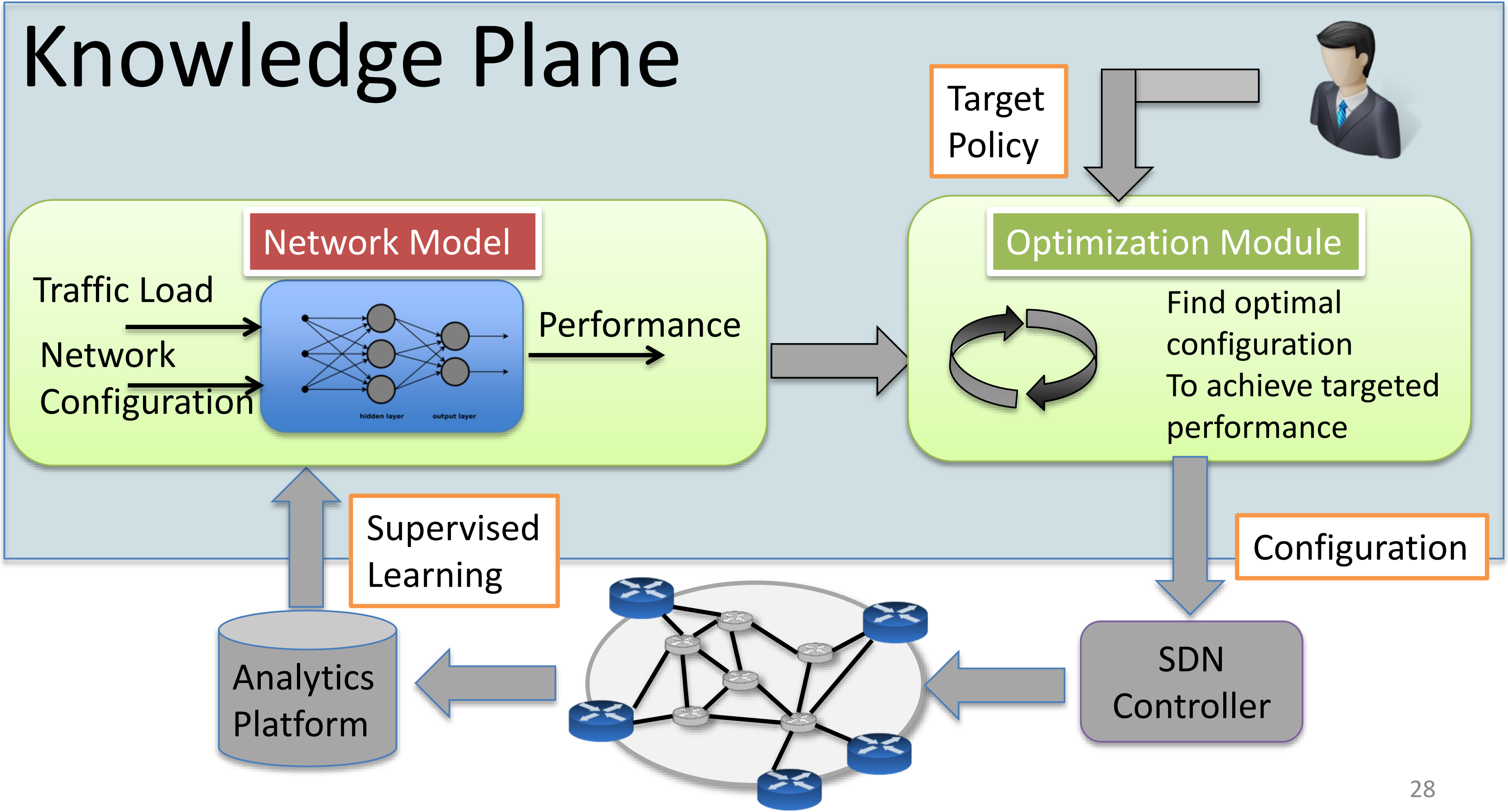
Advantages

- Accurate
- Can model the system as a black-box
- Scales well with complexity
- Trained models are very lightweight

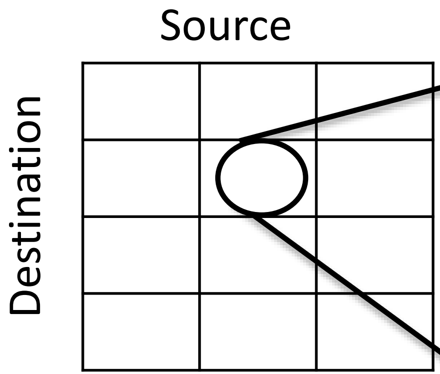
Challenges

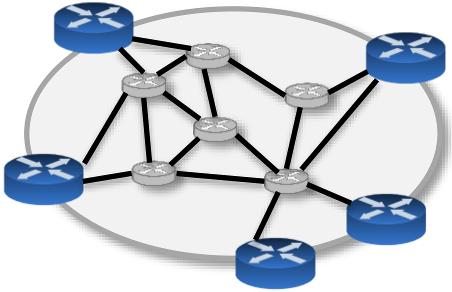
- Representative Dataset
- No guarantees
- Cannot be understood by humans

Optimization Module


Knowledge Plane
Target Policy
Network Model
Traffic Load
Network Configuration
hidden layer
output layer
Performance
Optimization Module
Find optimal configuration
To achieve targeted performance
Supervised Learning
Configuration
Analytics Platform
SDN Controller

How do you represent traffic load?



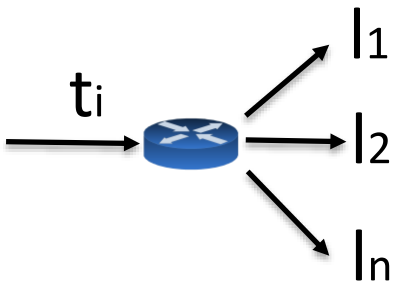


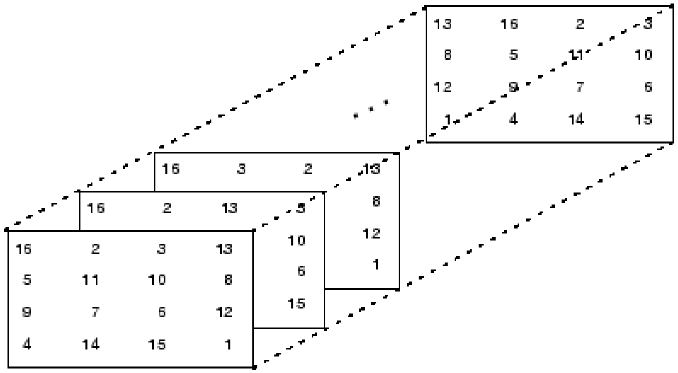
- Type of traffic
  - Destination-based flows
  - Traditional 5-tuple flows
  - 34-field type flows
  - …
- Each type of traffic is represented by a set of feature
  - E.g., Moments of the PDF of the inter-arrival time
  - E.g., Bandwdith, # Packets, etc
  - …
- **T** represents the Traffic Load as a multi-dimensional matrix

Example:

In a network with 20 ingress/egress nodes, assuming 10 types of traffic and 70 features per type: 280.000 elements

How do you represent the network configuration?





- For each type of traffic
  - Traditional 5-tuple flows
  - 34-field type flows
  - …
- Send the traffic through links l1..ln with a certain percentage of traffic per link
- **R** represents the Network Configuration as a multi-dimensional matrix

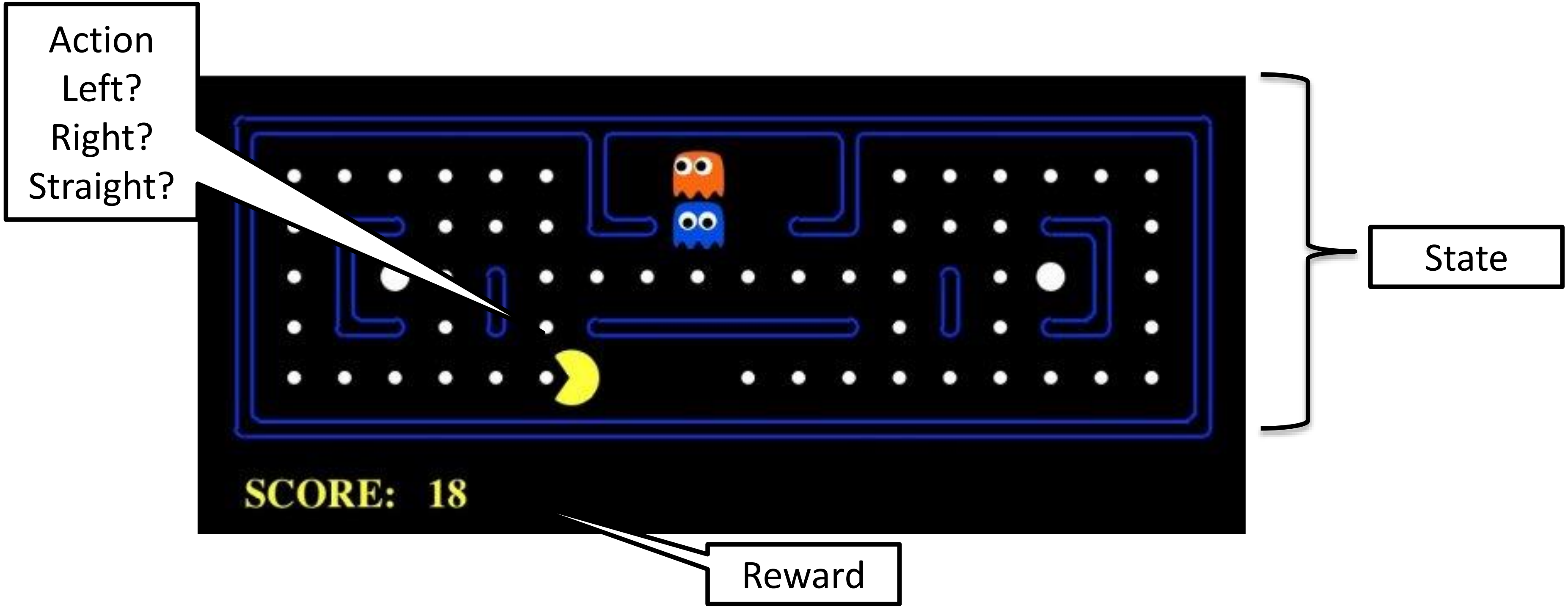
Example:

In a 100-sized network, assuming 10 types of traffic and 3 links per node: 300.000 elements.

Challenges in the Optimization

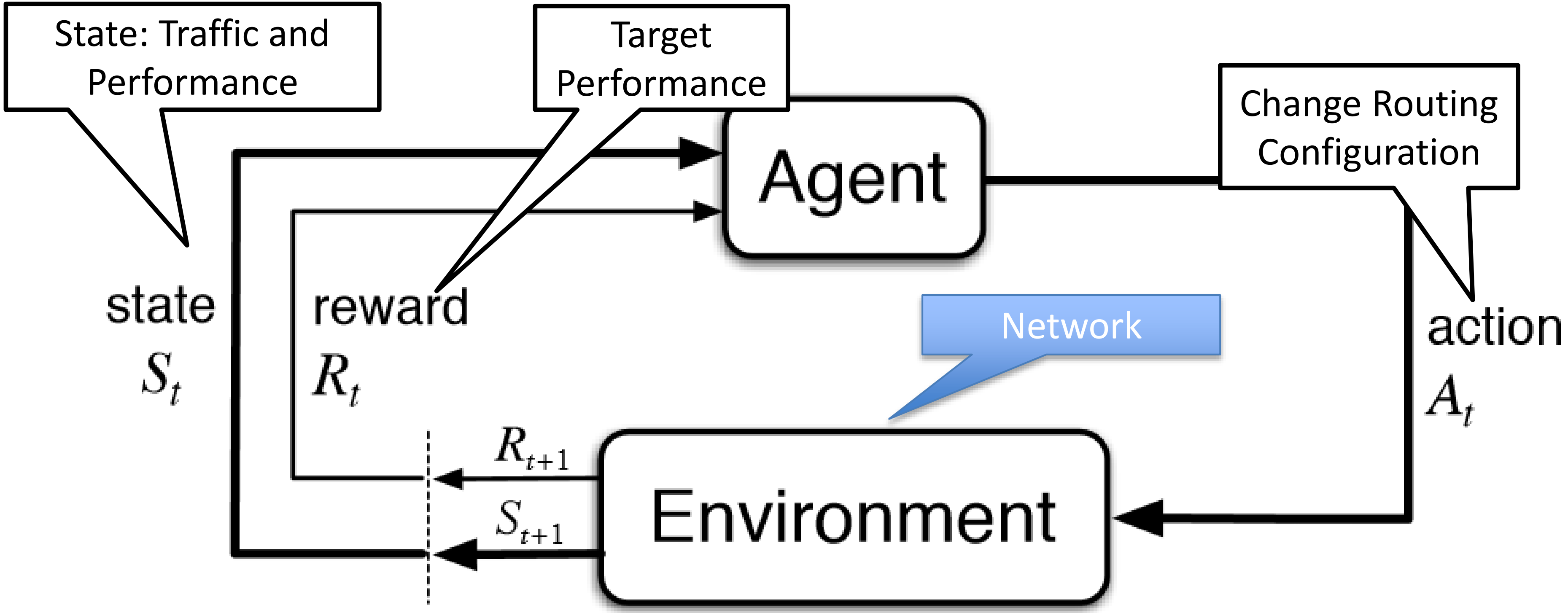
- Goal: Given the current traffic load `T` search for the network configuration `R` that achieves the target performance
- Space is very large, traditional optimization algorithms may be too slow

Reinforcement Learning

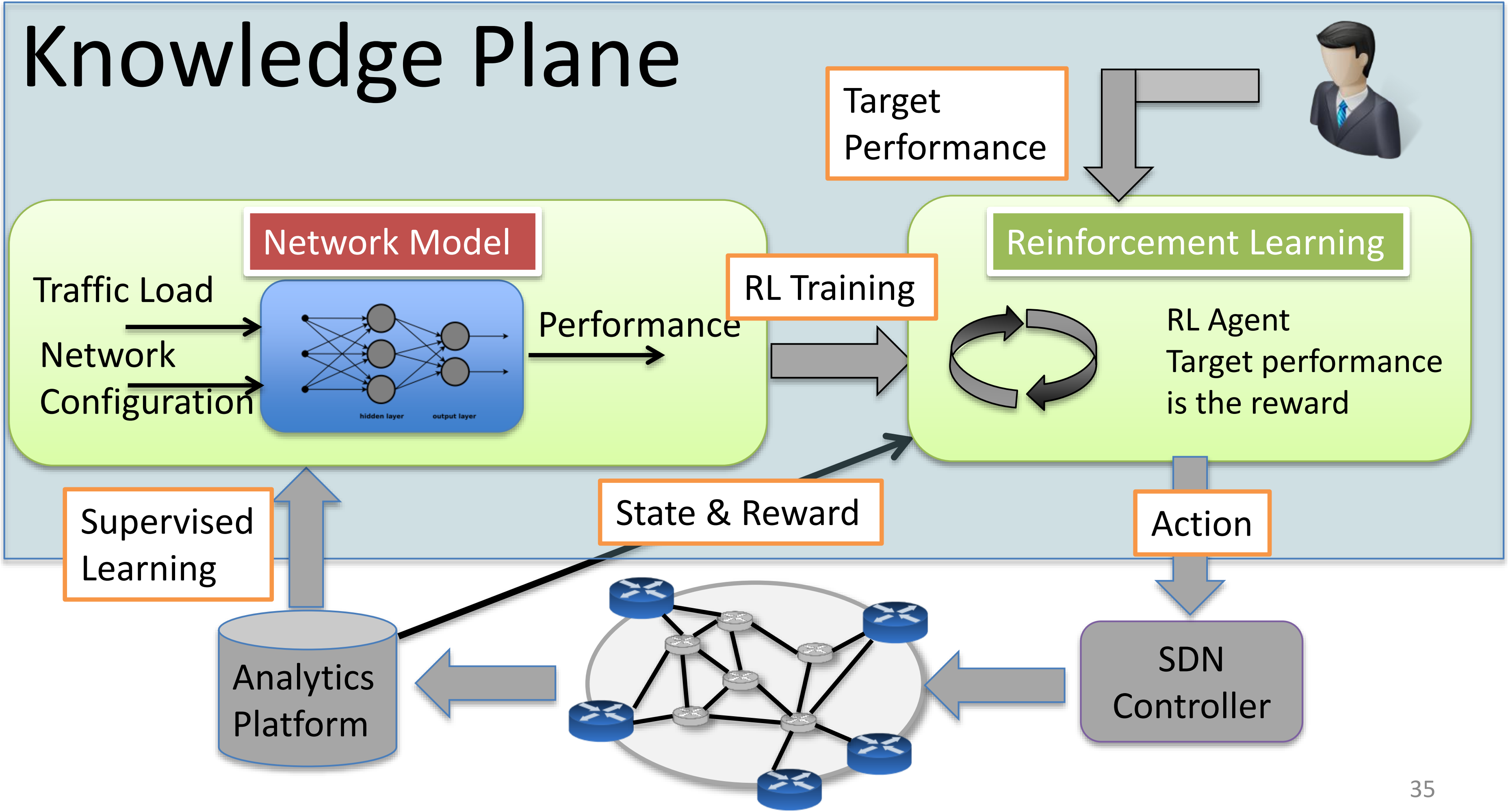


Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." *Nature* 518.7540 (2015): 529-533.

Reinforcement Learning



Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." *Nature* 518.7540 (2015): 529-533.


Knowledge Plane
Target
Performance
Network Model
Traffic Load
Network
Configuration
hidden layer
output layer
Performance
RL Training
Reinforcement Learning
RL Agent
Target performance
is the reward
Supervised
Learning
State & Reward
Action
Analytics
Platform
SDN
Controller

Challenges

- **`T`** and **`R`** are also too large to be fed to a Neural Network
  - *The curse of dimensionality*
- Potential Solutions
  - Feature Extraction
  - Autoencoders
  - Deep Reinforcement Learning

Feature Extraction

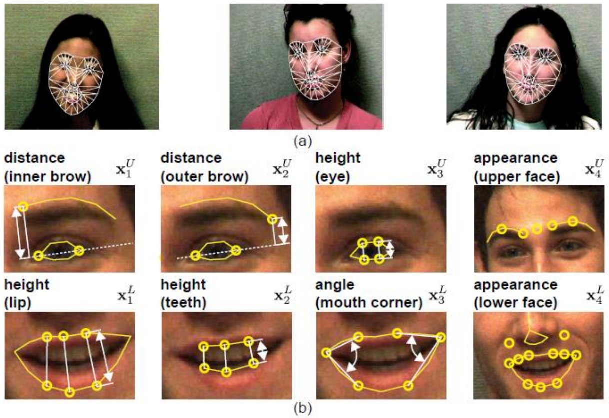
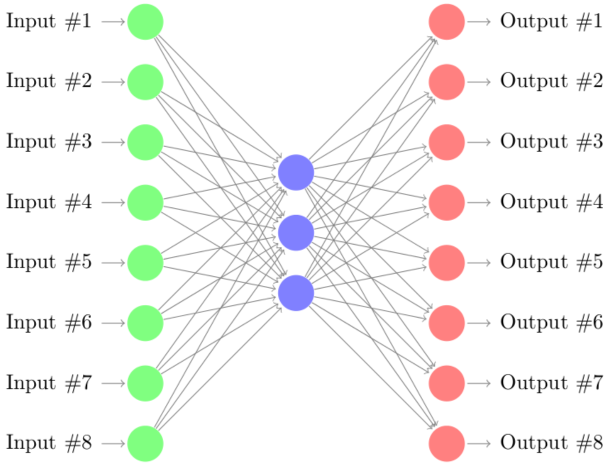


Figure 2: Facial features used for temporal clustering. (a) AAM fitting across different subjects. (b) Eight different features extracted from distance between tracked points, height of facial parts, angles for mouth corners, and appearance patches.

- Manual definition of features
- Example:
  - IGP Cost
- But what about the topology?
  - Traditional adjacency matrix do not work
- Advantages
  - Features have meaning to humans
  - System can be understood and troubleshoot to a certain extent
  - Can help provide some performance guarantees

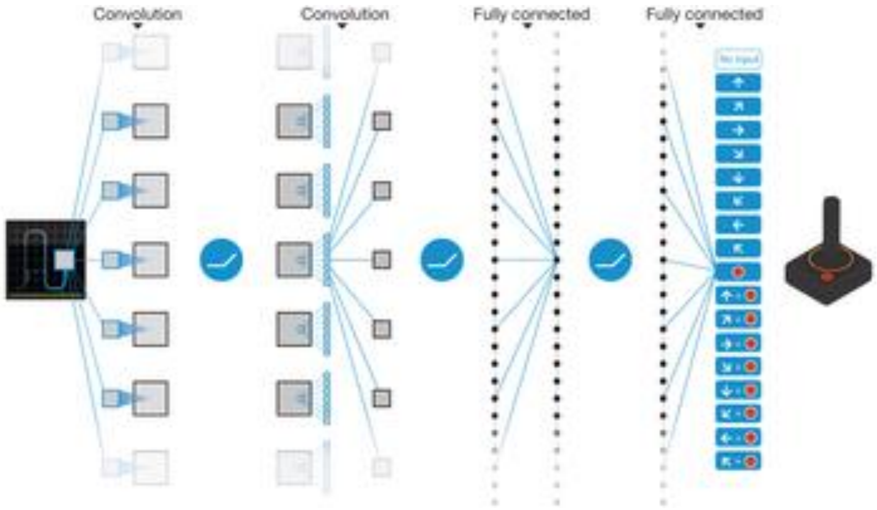
Kar, Abhishek. "Unsupervised temporal segmentation of facial behaviour."

Autoencoders



- Goal: Reduce dimensionality
- Can be understood as *automatic* feature engineering
- It works if there are correlations in the data
- Advantages
  - Automatic
- Disadvantages
  - Features do not have meaning for humans

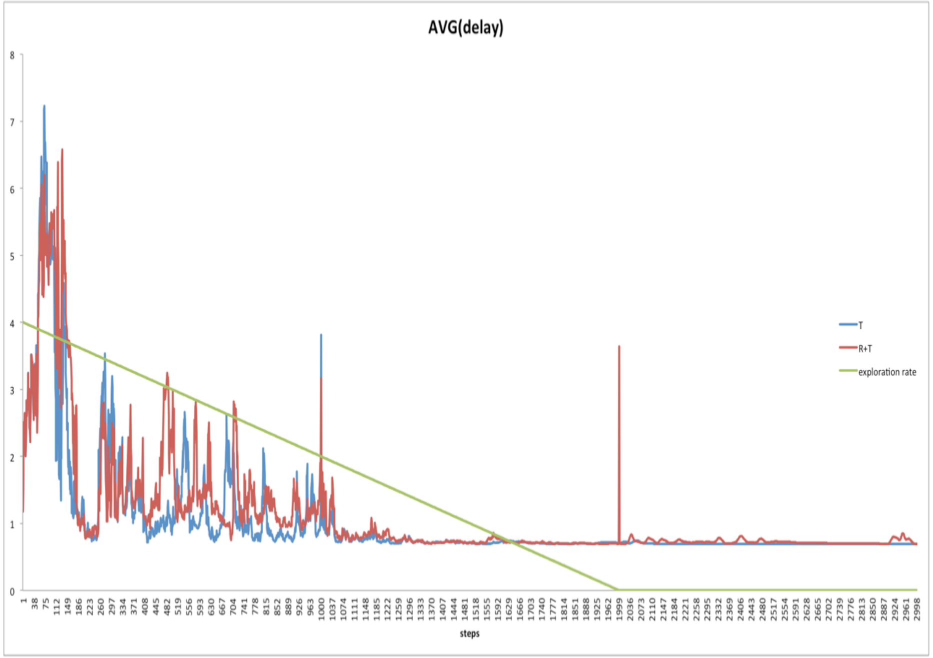
Deep Reinforcement Learning

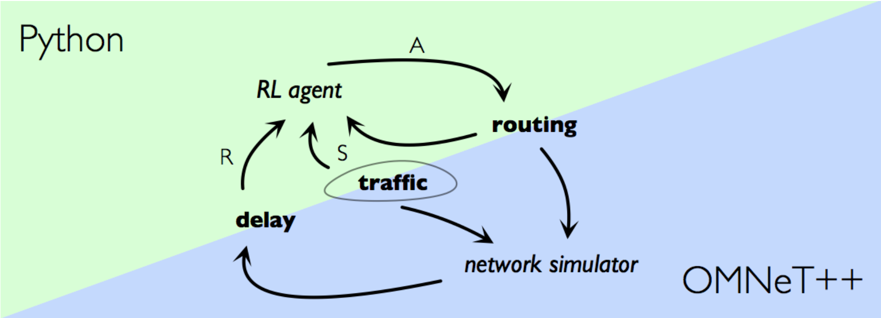


- Just use more neurons
  - Deep Neural Networks
- Advantages
  - Black-box approach
- Disadvantages
  - Costly training
  - The system cannot be understood by humans

Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." *Nature* 518.7540 (2015): 529-533.

Deep Reinforcement Learning: Preliminary Results





Note that this experiments needs to be scaled up

Conclusions & Future Work

Conclusions & Future Work

- Machine Learning represents a tool for network modeling that will result in unprecedented automation and optimization
- Advantages
  - Scales very well with complexity
  - Can understand the system as a black-box
  - Once trained, very lightweight
- Challenges
  - What is a representative data-set?
  - How do we represent fundamental network characteristics?
  - ML produces systems that do not offer guarantees and are hard to understand/troubleshoot by humans

Knowledge-Defined Networking

https://arxiv.org/pdf/1606.06222.pdf

Knowledge-Defined Networking

Albert Mestres*, Alberto Rodriguez-Natal*, Josep Carner*, Pere Barlet-Ros*, Eduard Alarcón*,
Marc Solé¶, Victor Muntés-Mulero¶,
David Meyer†, Sharon Barkai‡, Mike J Hibbett‖, Giovani Estrada‖, Florin Coras§,
Vina Ermagan§, Hugo Latapie§, Chris Cassar§, John Evans§, Fabio Maino§, Jean Walrand** and Albert Cabellos*
* Universitat Politècnica de Catalunya † Brocade Communication ‡ Hewlett Packard Enterprise
§ Cisco Systems ¶ CA Technologies ‖ Intel R&D ** University of California, Berkeley

***Abstract*—The research community has considered the application of Artificial Intelligence (AI) techniques to control and operate networks. A notable example is the Knowledge Plane as proposed by D.Clark et al. Such techniques have not, as yet, been extensively prototyped or deployed in the field. In this paper, we explore the reasons for the lack of adoption and posit that the rise of two recent paradigms: Software Defined Networking (SDN) and Network Analytics (NA), will facilitate the adoption of AI techniques in the context of network operation and control. We describe in some detail an architecture which accommodates and exploits SDN, NA and AI, and provide use cases that illustrate its applicability and benefits, together with simple experimental results that support its feasibility. We refer to this architecture as Knowledge Defined Networking (KDN).**

Keywords*—Knowledge Plane, SDN, Network Analytics, Machine Learning, NFV, Knowledge-Defined Networking

I. Introduction

D. Clark et al. proposed “A Knowledge Plane for the Internet” [1], a new construct that relies on Machine Learn-

techniques provide real-time, packet and flow-granularity information, as well as configuration and network state monitoring data to a centralized Network Analytics (NA) platform [4]. In this context, telemetry and analytics technologies provide a richer view of the network compared to what was possible with conventional network management approaches.

In this paper, we advocate that the centralized control offered by SDN, combined with a rich centralized view of the network provided by network analytics, enable the deployment of the KP concept proposed in [1]. In this context, the KP can use ML to gather knowledge about the network, and exploit that knowledge to control the network using logically centralized control facilities provided by SDN. We refer to the architecture resulting from combining SDN, telemetry, Network Analytics, and the Knowledge Plane as Knowledge-Defined Networking.

This paper first describes the Knowledge-Defined Networking (KDN) architecture and how it operates. Then, it describes a set of relevant use-cases that show the applicability of such architecture to networking and the benefits associated to using ML. In addition, for some use-cases, we also provide early