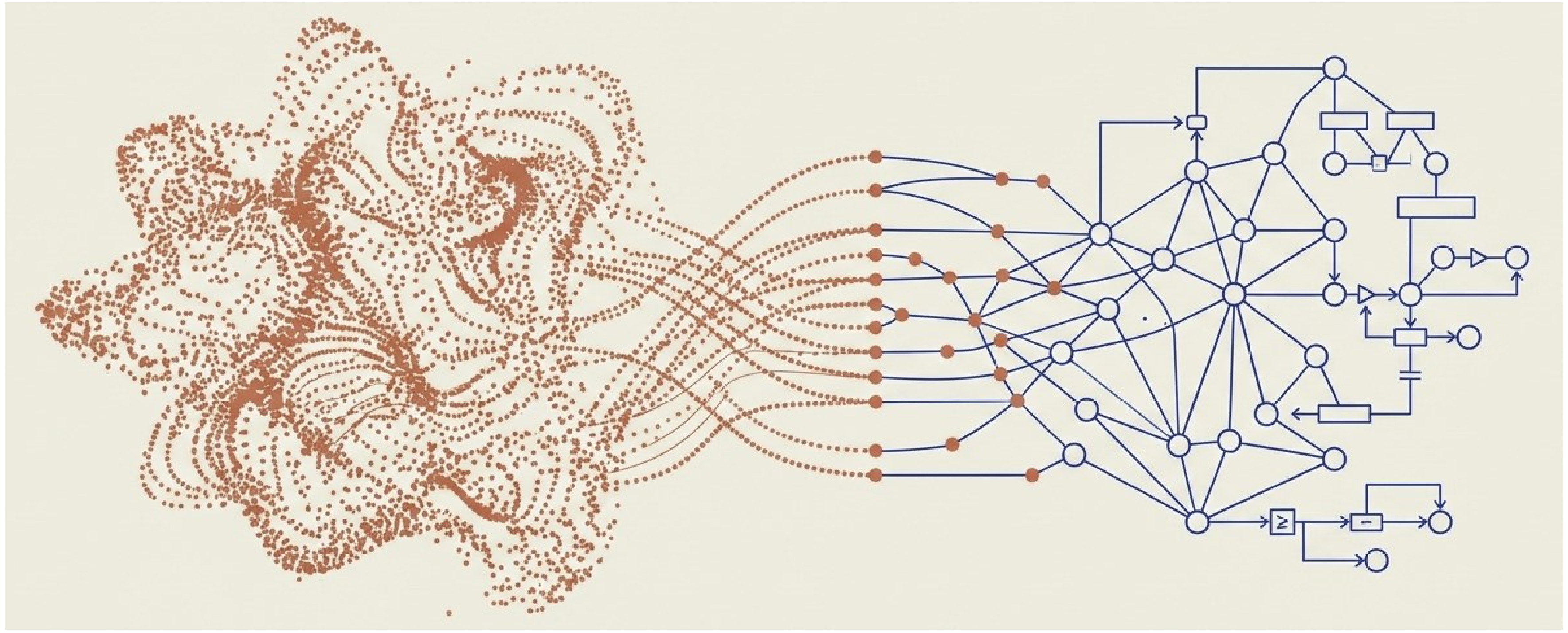
https://www.snap4City.org

NeuroSymbolic Artificial Intelligence at Scale

Parte: 1 (2025-26) What is Symbolic, Classification

Paolo Nesi, paolo.nesi@unifi.it

https://www.disit.org/



Structure of the course up to now …..

- Overview
- What is Symbolic, Classification ←
- Hybrid solutions
- Physically Informed, PINN
- Deep Reinforced Learning and Symbolic at Scale
- Knowledge ←→ NN
- From RAG LLM to Agentic LLM
- MLOps at Scale

What is «neuro» in Neuro-Symbolic

- **All kinds of Neural Networks**.

- What is not NN! We use a lot of non NN:
  - **Clustering**, which are typically non supervised → ML, no symbolic
  - **Decision Trees** as: RF, XGBoost → ML, no symbolic reasoning
  - **Regressions**: linear, multilinear, polinomial, ….
  - Ontologies and inferential rules
  - Etc.

What is Symbolic

- Represents a problem using **explicit symbols, variables, and logical/mathematical constraints**, to
  - solve/model the problem via **exact inference or optimization**, *not gradient descent.*
- **Variables**: Boolean, Integer, Real, Structured
- **Constraints**: logic / algebra
- **Objectives**:
  - Verify a condition
  - Min/max a functional

$$\min_x \; f(x) \quad \text{s.t.} \quad \bigwedge_{i=1}^{m} C_i(x)$$

Symbolic: Logic

- **SAT**, Boolean SATisfiability, **computing Boolean condition**:
  - E.g., C1: (A or B) and (not A or C); C2: …………… $(A \lor B) \land (\neg A \lor C)$
  - conditions, conjunctions sequentially ordered, etc.

- **SMT** (Satisfiability Module Theories) solving, **constraint programming**
  - *Boolean logic plus numbers*, arrays, bit operations, etc.
  - Computing: $(x>3) \land (y=x+2) \land (y<6)$

Symbolic: Logic

- **Propositional Logic, PL:**
  - When G is true ? [defined as G = C1: (A or B) and (not A or C)]
- **First-order logic** reasoning, description logics, add to PL:
  - universal quantification $\forall$, for each;
  - existentional quantification $\exists$, exists;
  - E.g.: does not exists a set x such that all other sets y are its element

$$\neg\exists x\ \forall y\ (y \in x)$$

- **Higher order logic:** quantifications over functions/pred.
  - $\forall f\ \forall x\ P(f(x))$
  - Theorem proving → Isabelle/HOL

What is Symbolic

- **Inductive Logic Programming** (ILP), rule learning
  - a machine learning approach that **learns logical rules (a program)** from examples, using **logic programming** (usually Prolog-style) and **background knowledge, B**

- **See Knowledge Engineering Course**

Inferential Rule

- **Facts, *B:***
  - parent(alice, bob). *// Alice è padre/madre di Bob*
  - parent(bob, carol). *// Bob è padre/madre di Carol*
  - parent(alice, dave)
- **Inferential Rule *H:***
  - "X is a grandparent of Z if X is a parent of Y and Y is a parent of Z."
- **Learned** via ***B U H***
  - grandparent(alice, carol) *// Bob è un/a nonno/a di Carol*

What is Symbolic

- **Expert systems** are AI programs that may simulate decision making in specific domains, and are grounded on:
  – Knowledge base (ontologies/rules plus instances/facts)
  – Inference engine
- **ES aims to produce**: suggestions, prescriptions, explanations

- **ES:** a *type* of DSS that mainly uses **rules/logic** for reasoning.
  – In the **course of Big Data Architectures and in this course**

DSS: Decision Support System

- **DSS should**
  - **supports semi-structured or unstructured decisions** (not just routine automation),
  - offers analysis, **recommendations**, or **what-if via simulation**,
  - keeps a human in the loop (the **user decides**),
  - **Explainability**: justify outputs with evidence, assumptions, and scenarios.

Classification of DSS

- **Data-driven DSS**
  - querying/visualizing large datasets
  - Examples: BI dashboards, KPI monitoring, drill-down analytics, anomaly alerts.
- **Model-driven DSS**
  - mathematical/optimization/simulation models
  - Examples: pricing optimization, supply-chain planning, portfolio optimization, scheduling, Monte Carlo risk.
- **Knowledge-driven DSS (includes many “expert systems”)**
  - rules/logic/knowledge bases
  - Examples: eligibility/coverage decision support, clinical guideline support, compliance rule engines.
- **Document-driven DSS → RAG LLM, Agentic LLM**
  - retrieving and synthesizing information from documents
  - Examples: legal discovery tools, policy lookup systems, RAG-based “answer with citations.”
- **Communication/Collaboration-driven DSS**
  - supporting group decisions
  - Examples: groupware for prioritization, meeting decision tools, consensus/workflow platforms.

Classifying Decision Support Systems (DSS) in Smart Cities

1. Data-Driven DSS

Focuses on querying and visualizing large datasets to find immediate insights.

- Business Intelligence (BI) dashboards
- KPI monitoring & drill-down analytics
- Real-time anomaly alerts

2. Model-Driven DSS

Focuses on mathematical, optimization, and simulation models to project outcomes.

- Pricing & supply-chain optimization
- Traffic flow simulation
- Monte Carlo risk assessment

3. Knowledge-Driven DSS

Focuses on symbolic rules, logic, and expert systems to ensure compliance.

- Regulatory compliance rule engines
- Clinical guideline support
- Automated expert reasoning

Modern AI-enabled DSS (today's common pattern)

- **Predictive DSS:**
  - – ML predicts outcomes (churn, default risk, demand) + gives confidence intervals.
- **Prescriptive DSS:**
  - – recommends actions (optimize, allocate, schedule) often combining ML + optimization.
- **Agentic DSS:**
  - – an LLM agent that gathers info, runs tools (SQL, solver, simulator), proposes options, and produces a traceable report.

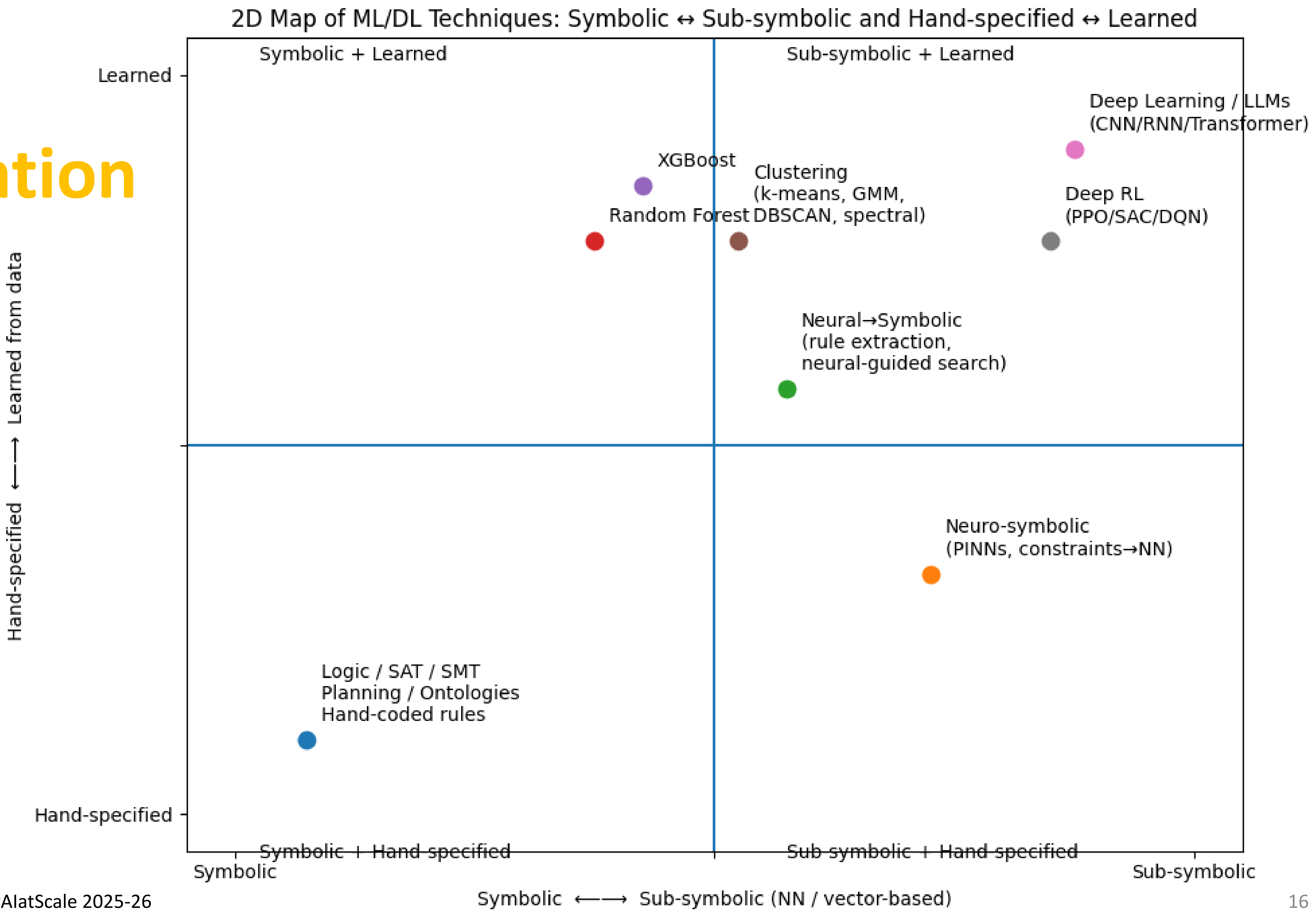
Symbolic vs Sub-Symbolic

- **Symbolic**:
  - The meaning is **explicit** in symbols;
  - reasoning is discrete and traceable (explainable).
- **Sub-symbolic:**
  - The meaning is **implicit** in learned numeric representations;
  - "reasoning" is emergent from computation.
  - **Note:** belong to this category:
    - Neural Networks
    - Probabilistic models

UNIVERSITÀ DEGLI STUDI FIRENZE

DI DIPART INGEGN DELL'IN

Classification



- **Tree models** (RF/XGBoost)
  - have **discrete structure** (symbolic-ish)
  - but are **trained statistically** (not symbolic reasoning).

- **Neuro-symbolic systems** mix both: e.g.
  - **PINNs** (physics constraints + NN),
  - **LLM agents calling solvers/planners**, or
  - **RL with symbolic safety shields**.

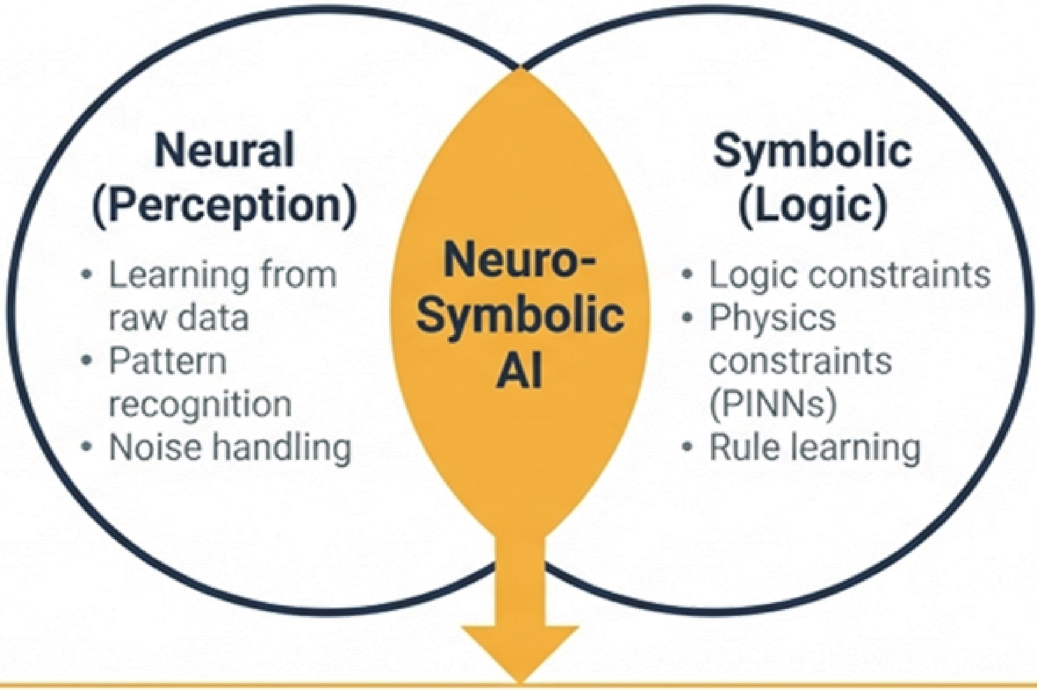
Symbolic (explicit symbols + discrete reasoning)

- **What it is**
  - Knowledge as human-interpretable symbols: rules, facts, logic formulas, programs, graphs/ontologies.
  - Inference by explicit operations on symbols: matching rules, search, planning, SAT/SMT solving.
- **strengths**
  - Interpretability & traceability: you can inspect “why” step-by-step.
  - Compositionality: easy to combine pieces (rules + rules).
  - Constraints/guarantees: you can enforce safety/specs (e.g., “never violate constraint X”) more directly.
- **weaknesses**
  - Fragility: fails if the world doesn’t match the symbols exactly, non decidable, non complete.
  - Knowledge engineering cost: building/maintaining rules/ontologies can be expensive.
  - Harder to learn from raw data (images, audio) without a perception front-end.
  - Some Problems: executability, decidability, etc.
- **Examples**
  - Logic rules, theorem proving, classical planning, SAT/SMT constraint solving, ontologies.

Sub-symbolic (distributed numeric representations)

- **What it is**
  - Knowledge is encoded in continuous vectors/weights, not explicit symbols.
  - Decisions come from numeric computation (e.g., a neural network forward pass) learned from data.
- **strengths**
  - Perception & pattern recognition: great for images, speech, messy text.
  - Robustness to noise: can generalize from examples.
  - Scales well with data/compute.
- **weaknesses**
  - Opacity: “why” is not naturally a readable chain of steps.
  - Reliability: can be fooled by distribution shift; may hallucinate or behave inconsistently.
  - Harder to enforce strict constraints without extra machinery.
- **Examples**
  - Deep learning models (CNNs, Transformers/LLMs, diffusion), deep RL policies.

Bridging the Gap: The Neuro-Symbolic AI Approach



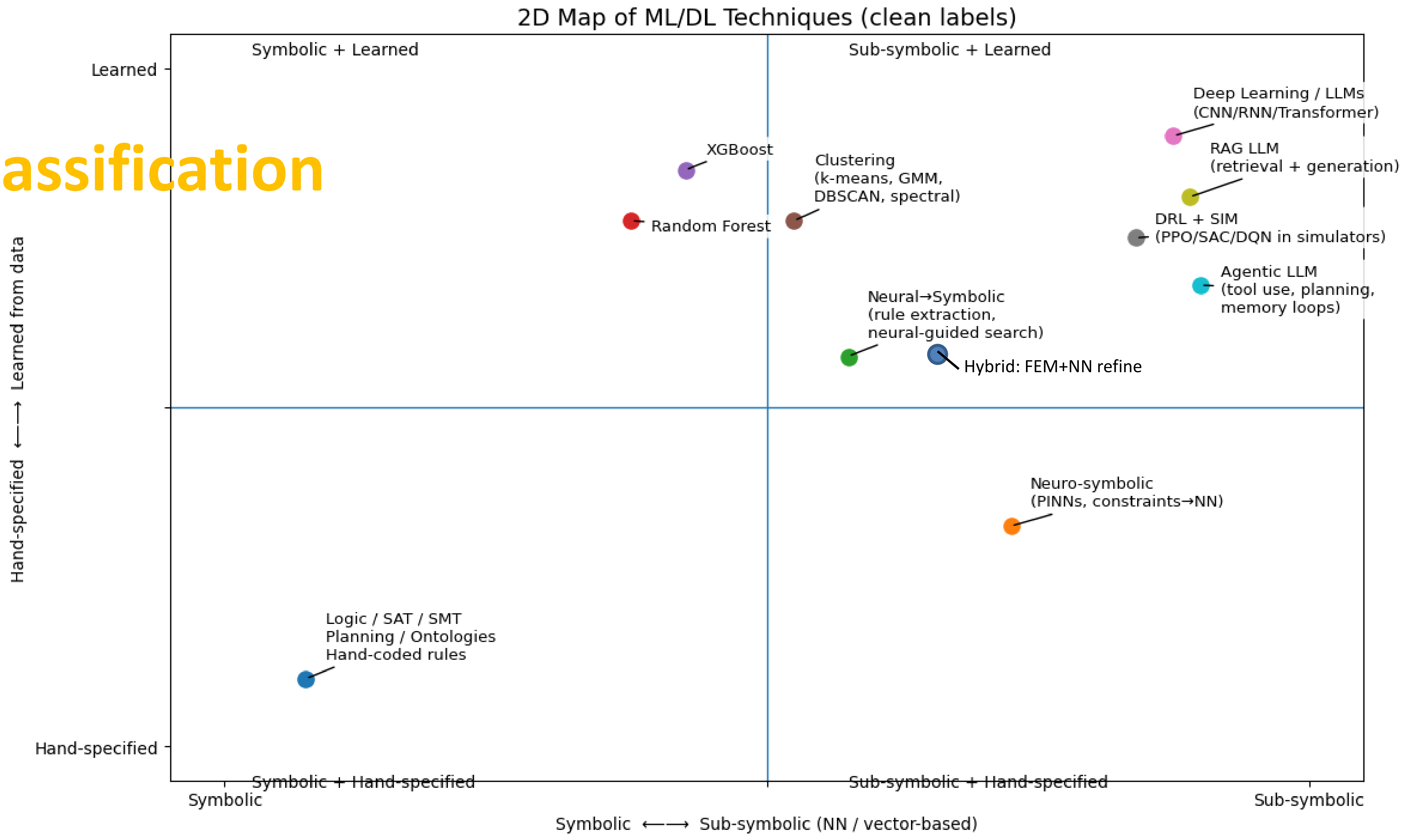
Key Outcomes: Deep Tight Inter Tight

- Extracts discrete structure from noisy urban data
- **Symbol Grounding**: Maps perception to actionable symbols
- Improved generalization and robustness
- **Explainable AI** (XAI) for transparent management

A Neuro-Symbolic Classification

- **Symbolic → Neural (knowledge as bias/constraints)** inject structure into learning:
  - Logic constraints as loss terms ("semantic loss" style)
  - Physics constraints (**PINNs**) and other mechanistic constraints
  - Type systems, ontologies, invariances, monotonicity constraints
  - → improved generalization, robustness, safety/spec compliance.
- **Neural → Symbolic (learning symbols/programs)** extract discrete structure
  - Rule learning / program induction from data
  - Symbol grounding: learn mappings from perception to symbols
  - Model distillation into rules/trees for interpretability
  - → interpretability, compositional generalization, verifiable reasoning.
- **Joint / co-trained**

Classification



Structure of the course up to now …..

- Overview
- What is Symbolic, Classification
- Hybrid solutions
- Physically Informed, PINN
- Deep Reinforced Learning and Symbolic at Scale
- Knowledge ←→ NN
- From RAG LLM to Agentic LLM
- MLOps at Scale

Vincolare le NN

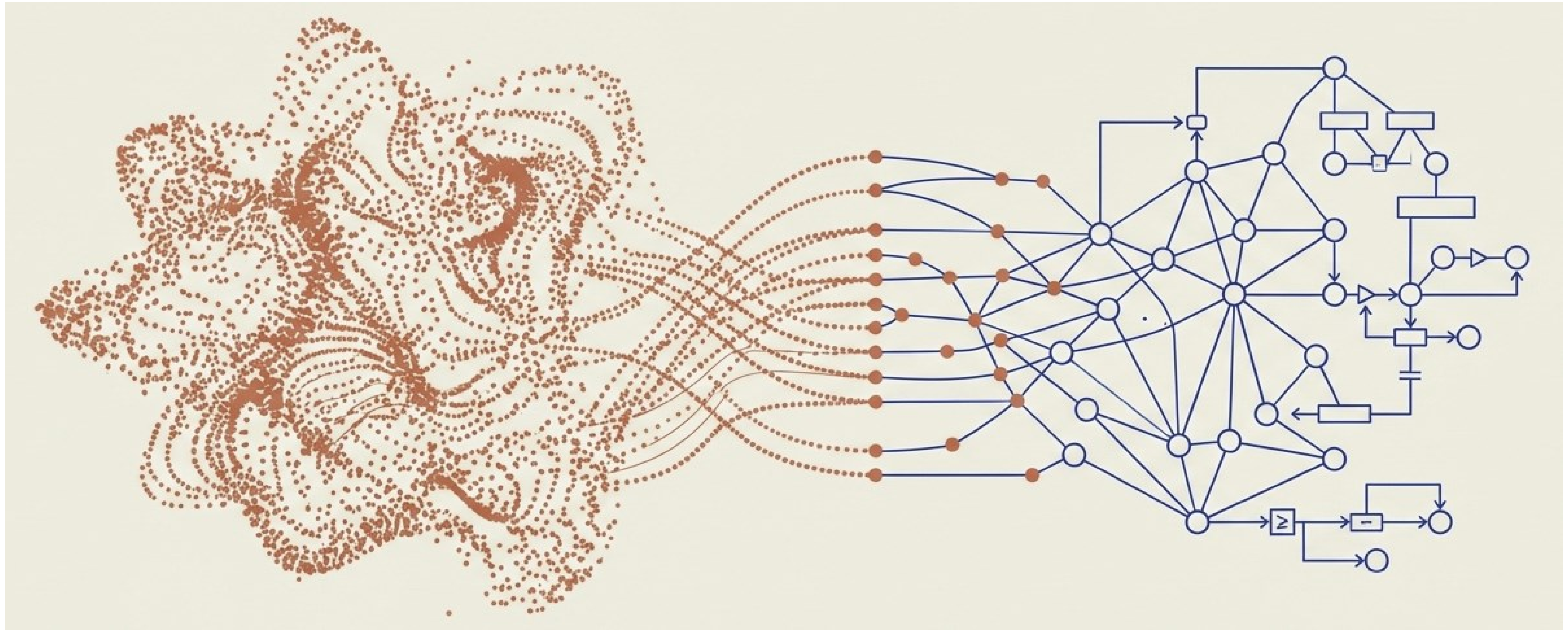
- Imporre regole decisionali o tenere conto di modelli simbolici alla rete tramite varie tecniche
- **Per esempio:**
  - – Soluzioni Ibride (refinement)
  - – Physically informed NN, PINN (loss)

NeuroSymbolic Artificial Intelligence at Scale

Parte: 2 (2025-26) Hybrid Solution

Paolo Nesi, paolo.nesi@unifi.it

https://www.disit.org/



An example of Hybrid Approach

- **hybrid / neuro-symbolic (physics-based + neural)** method
  - FEM + NN refinement of results

- **Finite Elements (FEM)** is physics-based, model-driven computation: it uses an explicit PDE + variational form + mesh/shape functions + solver.
  - it is explicit structured knowledge (equations + discretization)
- **The NN refinement is sub-symbolic / numerical**
  - learned continuous representation.

UNIVERSITÀ DEGLI STUDI FIRENZE

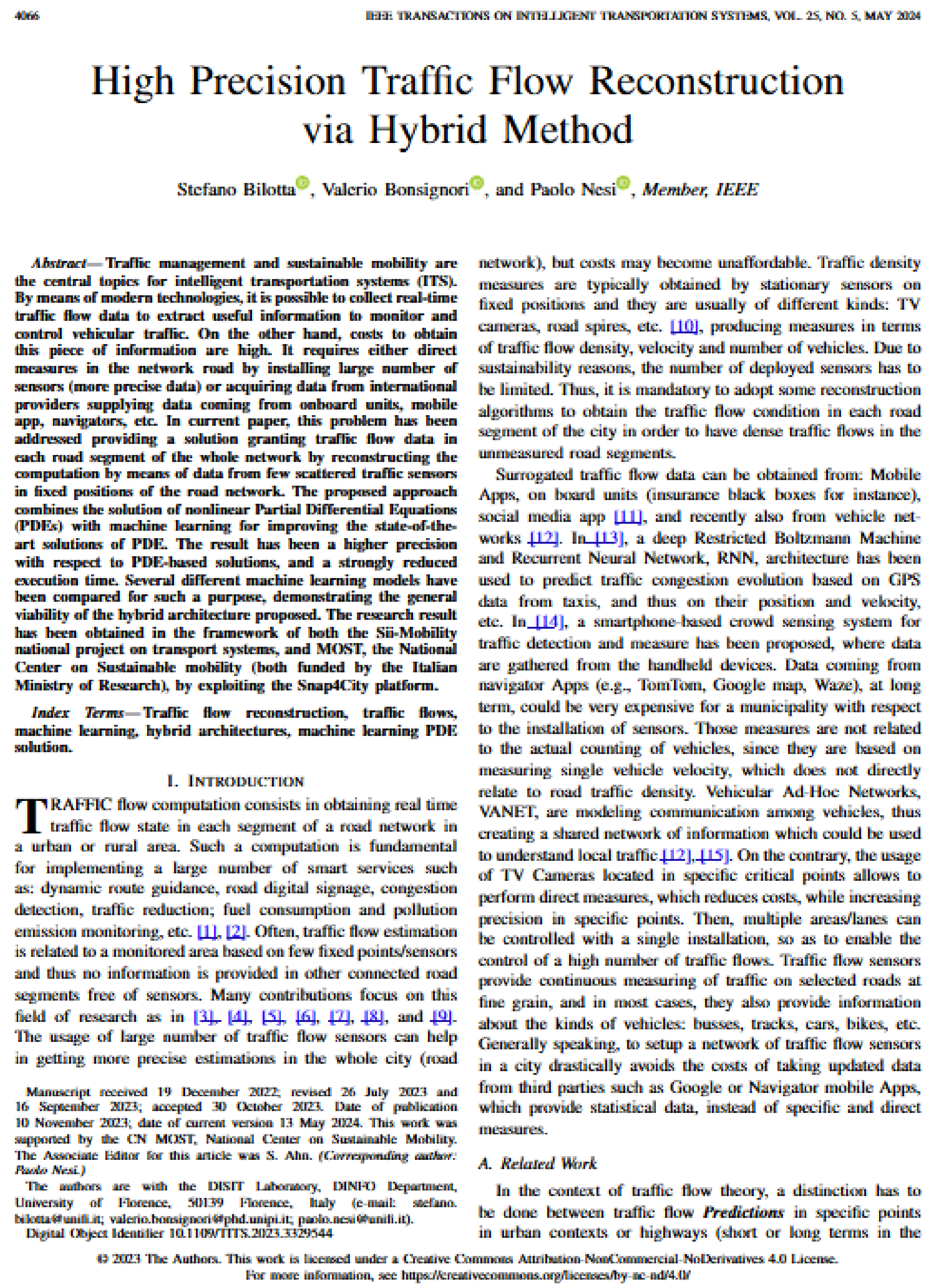
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB

- S. Bilotta, V. Bonsignori and P. Nesi, "High Precision Traffic Flow Reconstruction via Hybrid Method," in ***IEEE Transactions on Intelligent Transportation Systems***, vol. 25, no. 5, pp. 4066-4076, May 2024, doi: 10.1109/TITS.2023.3329544.

https://ieeexplore.ieee.org/document/10315063

Use a PDE-based reconstructor to generate dense traffic states, then train ML models to reproduce (and improve) reconstructions faster, handling missing data and adding temporal features.



4066 IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 25, NO. 5, MAY 2024

High Precision Traffic Flow Reconstruction via Hybrid Method

Stefano Bilotta, Valerio Bonsignori, and Paolo Nesi, *Member, IEEE*

***Abstract*—Traffic management and sustainable mobility are the central topics for intelligent transportation systems (ITS). By means of modern technologies, it is possible to collect real-time traffic flow data to extract useful information to monitor and control vehicular traffic. On the other hand, costs to obtain this piece of information are high. It requires either direct measures in the network road by installing large number of sensors (more precise data) or acquiring data from international providers supplying data coming from onboard units, mobile app, navigators, etc. In current paper, this problem has been addressed providing a solution granting traffic flow data in each road segment of the whole network by reconstructing the computation by means of data from few scattered traffic sensors in fixed positions of the road network. The proposed approach combines the solution of nonlinear Partial Differential Equations (PDEs) with machine learning for improving the state-of-the-art solutions of PDE. The result has been a higher precision with respect to PDE-based solutions, and a strongly reduced execution time. Several different machine learning models have been compared for such a purpose, demonstrating the general viability of the hybrid architecture proposed. The research result has been obtained in the framework of both the Sii-Mobility national project on transport systems, and MOST, the National Center on Sustainable mobility (both funded by the Italian Ministry of Research), by exploiting the Snap4City platform.**

***Index Terms*—Traffic flow reconstruction, traffic flows, machine learning, hybrid architectures, machine learning PDE solution.**

I. Introduction

TRAFFIC flow computation consists in obtaining real time traffic flow state in each segment of a road network in a urban or rural area. Such a computation is fundamental for implementing a large number of smart services such as: dynamic route guidance, road digital signage, congestion detection, traffic reduction; fuel consumption and pollution emission monitoring, etc. [1], [2]. Often, traffic flow estimation is related to a monitored area based on few fixed points/sensors and thus no information is provided in other connected road segments free of sensors. Many contributions focus on this field of research as in [3], [4], [5], [6], [7], [8], and [9]. The usage of large number of traffic flow sensors can help in getting more precise estimations in the whole city (road network), but costs may become unaffordable. Traffic density measures are typically obtained by stationary sensors on fixed positions and they are usually of different kinds: TV cameras, road spires, etc. [10], producing measures in terms of traffic flow density, velocity and number of vehicles. Due to sustainability reasons, the number of deployed sensors has to be limited. Thus, it is mandatory to adopt some reconstruction algorithms to obtain the traffic flow condition in each road segment of the city in order to have dense traffic flows in the unmeasured road segments.

Surrogated traffic flow data can be obtained from: Mobile Apps, on board units (insurance black boxes for instance), social media app [11], and recently also from vehicle networks [12]. In [13], a deep Restricted Boltzmann Machine and Recurrent Neural Network, RNN, architecture has been used to predict traffic congestion evolution based on GPS data from taxis, and thus on their position and velocity, etc. In [14], a smartphone-based crowd sensing system for traffic detection and measure has been proposed, where data are gathered from the handheld devices. Data coming from navigator Apps (e.g., TomTom, Google map, Waze), at long term, could be very expensive for a municipality with respect to the installation of sensors. Those measures are not related to the actual counting of vehicles, since they are based on measuring single vehicle velocity, which does not directly relate to road traffic density. Vehicular Ad-Hoc Networks, VANET, are modeling communication among vehicles, thus creating a shared network of information which could be used to understand local traffic [12], [15]. On the contrary, the usage of TV Cameras located in specific critical points allows to perform direct measures, which reduces costs, while increasing precision in specific points. Then, multiple areas/lanes can be controlled with a single installation, so as to enable the control of a high number of traffic flows. Traffic flow sensors provide continuous measuring of traffic on selected roads at fine grain, and in most cases, they also provide information about the kinds of vehicles: busses, tracks, cars, bikes, etc. Generally speaking, to setup a network of traffic flow sensors in a city drastically avoids the costs of taking updated data from third parties such as Google or Navigator mobile Apps, which provide statistical data, instead of specific and direct measures.

A. Related Work

In the context of traffic flow theory, a distinction has to be done between traffic flow ***Predictions*** in specific points in urban contexts or highways (short or long terms in the

Manuscript received 19 December 2022; revised 26 July 2023 and 16 September 2023; accepted 30 October 2023. Date of publication 10 November 2023; date of current version 13 May 2024. This work was supported by the CN MOST, National Center on Sustainable Mobility. The Associate Editor for this article was S. Ahn. *(Corresponding author: Paolo Nesi.)*

The authors are with the DISIT Laboratory, DINFO Department, University of Florence, 50139 Florence, Italy (e-mail: stefano.bilotta@unifi.it; valerio.bonsignori@phd.unipi.it; paolo.nesi@unifi.it).

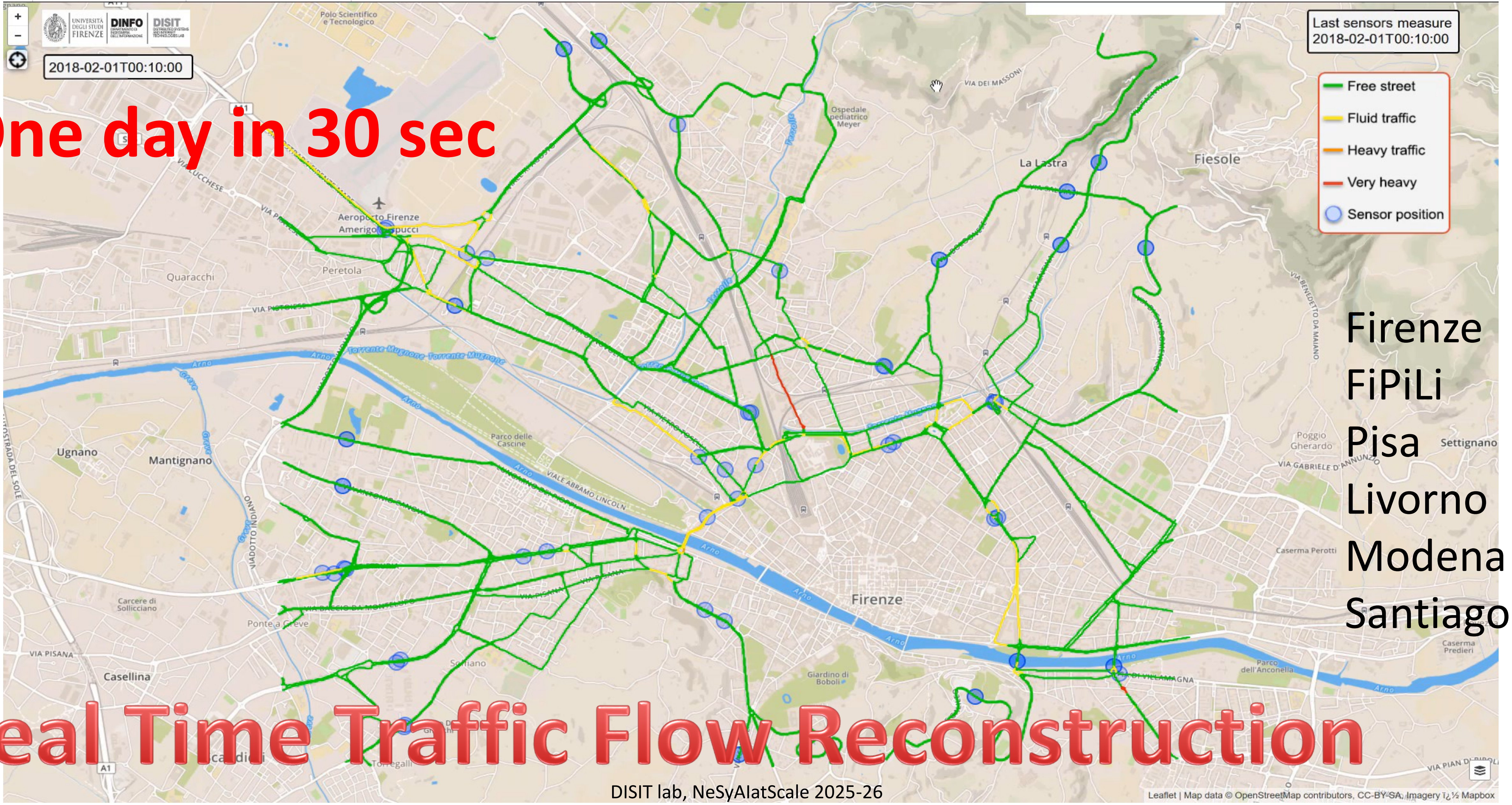
Digital Object Identifier 10.1109/TITS.2023.3329544

© 2023 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

High-Precision Traffic Flow Reconstruction via a Hybrid PDE + ML Method

- How the paper combines nonlinear PDE traffic modeling (LWR) with machine learning to improve accuracy and runtime
- Traffic Flow Reconstruction, TFR
  - Passing from scattered data to dense data on traffic flow solving PDE, + knowing data in a number of points, it is an ill-posed problems
  - solving indeterminacy at junctions, and adding other hyp/equations
    - At junctions, measured inflows do not uniquely determine outflows. Distribution patterns (turn ratios) vary with time and context.

One day in 30 sec



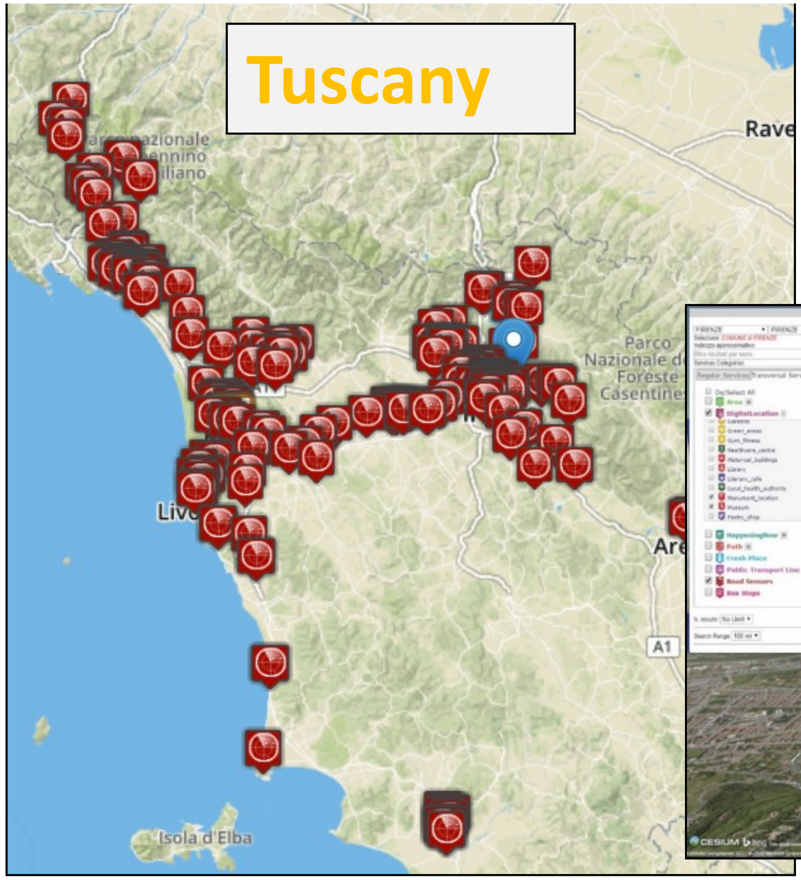
Firenze
FiPiLi
Pisa
Livorno
Modena
Santiago

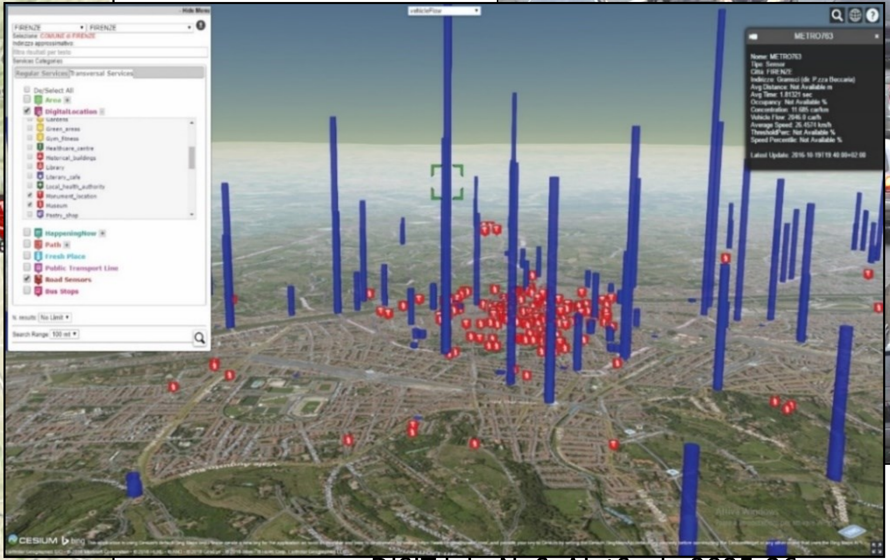
Real Time Traffic Flow Reconstruction

Traffic Flow Tools

Spire and Virtual Spires (cameras), Bluetooth, ...

Specifically located: along, around, on gates, on x…

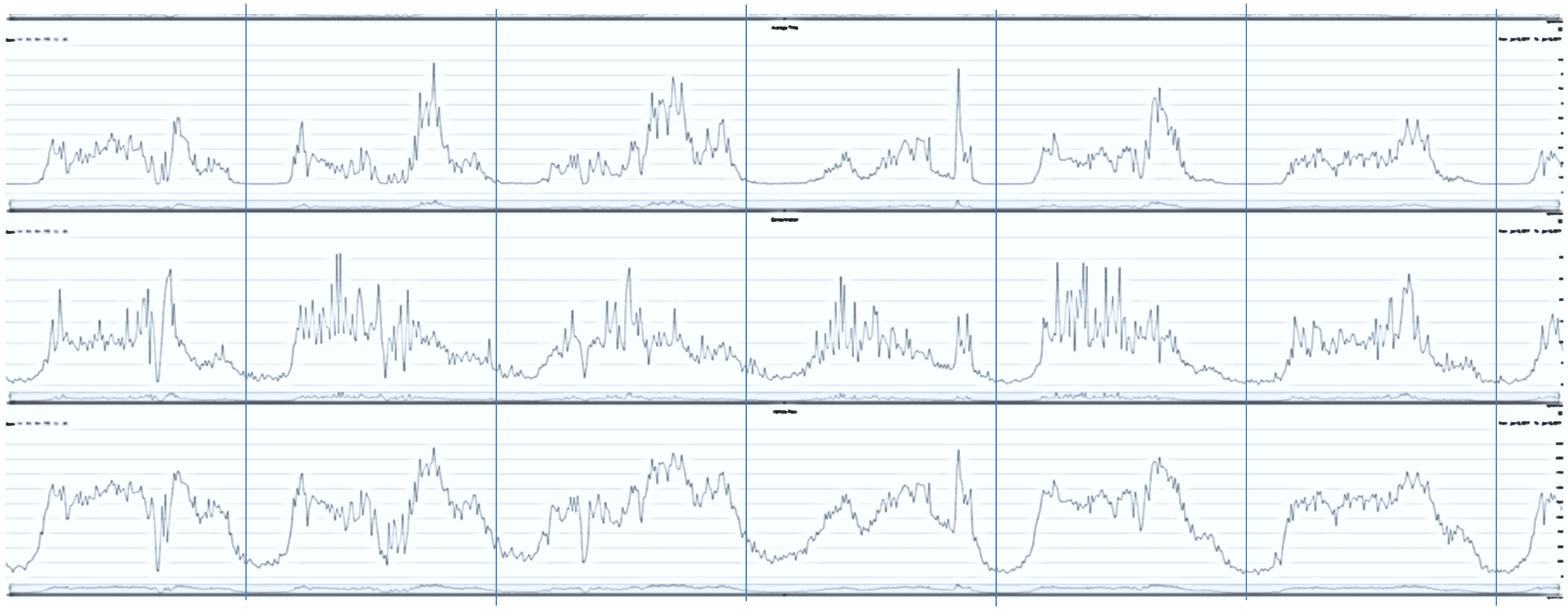




UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

Traffic Flow data

SNAP4CITY KM4CITY



- Day by day traffic flow, on the week data from 3 sensors

Motivation:

- Why reconstruct traffic flows for an entire road network?
  - **Traffic management needs dense, real-time network state — but sensors are sparse and expensive.**
- Installing many fixed sensors yields accurate data, but costs scale poorly.
- Third-party mobility providers can be costly and may not measure density directly.
- **Goal**: infer dense traffic density/flow on every road segment from a few sensors.
- **Constraints**: accuracy, runtime, and robustness to missing observations

Motivations / requirements 2

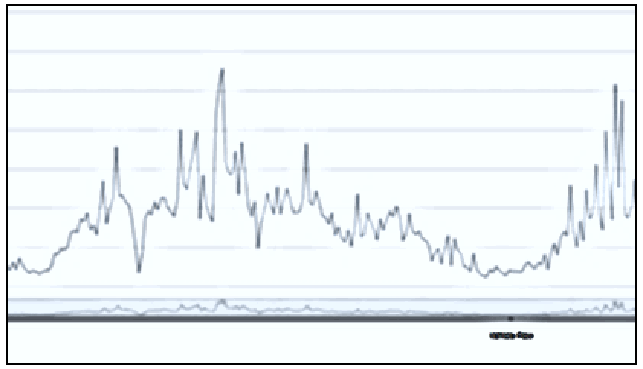
A full-network **Traffic Flow Reconstruction** (TFR):

- Passing from scattered data to dense flows
- density/flow per road segment (e.g., ~20m units)
- updated every sampling interval
- Indeterminacy flow distribution at junctions
- **Needed for:**
  – routing, dynamic routing, congestion monitoring, digital signage,
  – emissions estimation,
  – tuning semaphores, planning traffic flow infrastructure

Sensors and detections

- **Traffic Sensors static information** (identifier, geolocation, street address, technical specifications…) and the traffic flow detections (sensor, timestamp, detected traffic flow, estimated speed…) all come from publicly available Open Data.
  - Very noisy data
  - Missing data, samples
- **Sensors' data is** managed through data ingestion processes (ETL, IoT App), and stored in a No-SQL database, Big Data.
- **Sensors' data is** read every 10 minutes, the refresh frequency of the traffic sensors.
  - **Many variables including density, velocity, some KPI, aggregated and non-aggregated (equivalent vehicles)**
- The **TFR model process implementation** accesses those data through dedicated APIs, and **computes in real time**.

Aeroporto Firenze
Amerigo Vespucci
etola



Mathematical model

- **Traffic sensors** are interpreted as *sources of traffic* leading into the outcoming roads of the nodes where sensors are located.
- **A mathematical model for fluid dynamic flows** on networks which **is based on conservation laws.**
- **Road network** is a directed graph composed by arcs that meet at some nodes, the junctions.

Mathematical model

- **A fluid-dynamic model** can be seen as a macroscopic model which allows to observe the network in the time evolution through waves formation. **Propagation back and forward**
- Hyp.: a nonlinear model based on the conservation of cars described by the following scalar hyperbolic conservation law (1)

$$\frac{\partial \rho(t,x)}{\partial t} + \frac{\partial f\left(\rho(t,x)\right)}{\partial x} = 0$$

$\rho(t,a) = \rho_a(t)$ and $\rho(t,b) = \rho_b(t), \;\; \rho(0,x) = \rho_0(x).$

Mathematical model

- $\boldsymbol{\rho}(\boldsymbol{t}, \boldsymbol{x})$ denotes the **car density** which admits values from 0 to $\rho_{\max}$, where $\rho_{\max} > 0$ is the maximal vehicular density on the road.
- $\boldsymbol{f}(\boldsymbol{\rho}(\boldsymbol{t}, \boldsymbol{x}))$ is **the vehicular flux** which is defined as the product $\rho(t, x)v(t, x)$, where $v(t, x)$ is the local speed of the cars.
- In first order approximation, we assume that $v(t, x)$ is a decreasing function, only depending on the density, then the corresponding flux is a concave function

Fundamental diagram

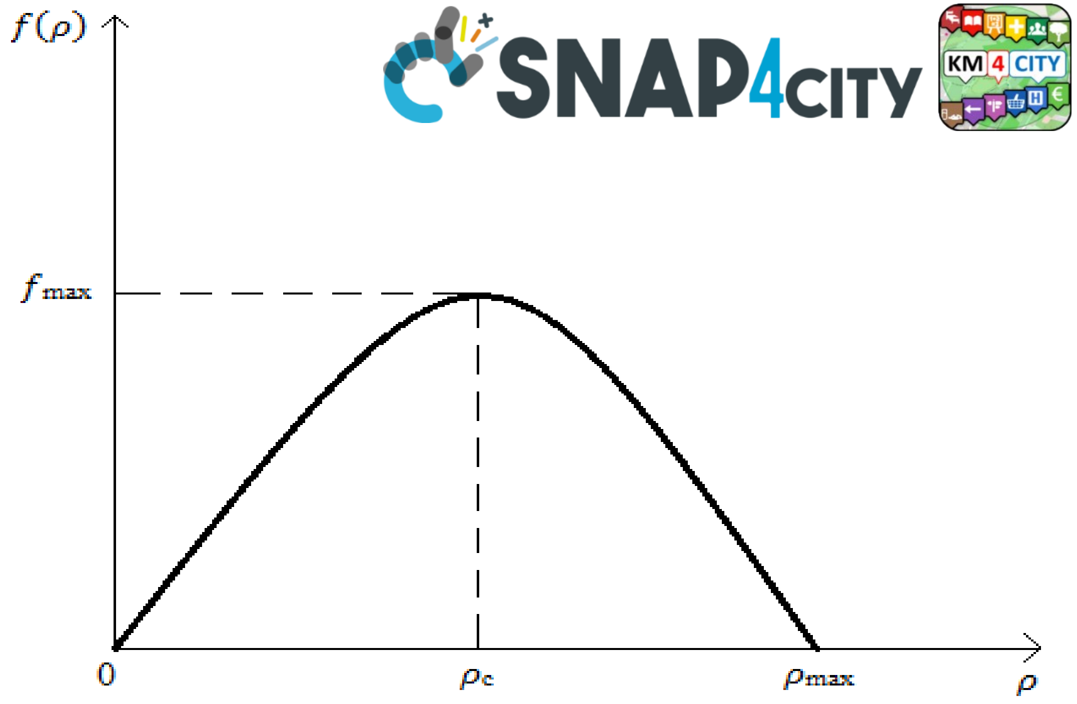
We consider the local cars' speed as

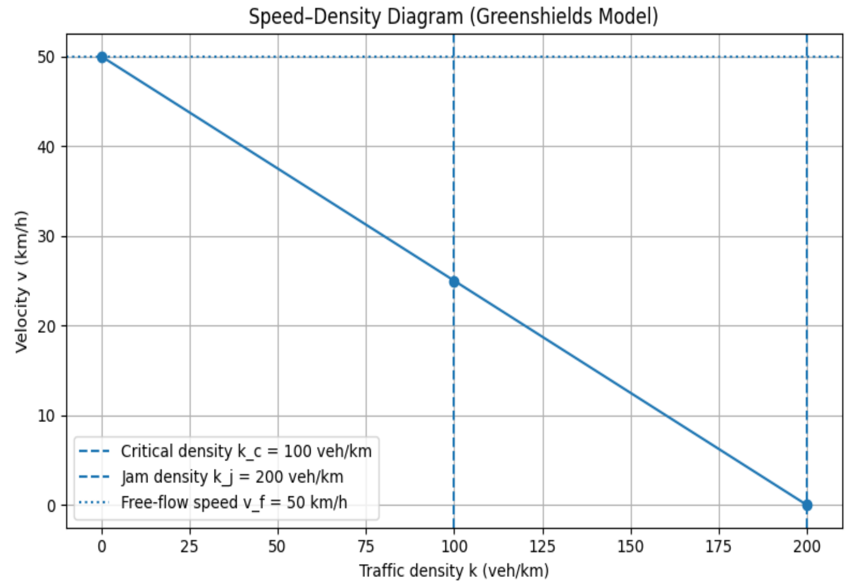
$$v(\rho) = v_{max}(1 - \frac{\rho}{\rho_{max}})$$

obtaining that the flow

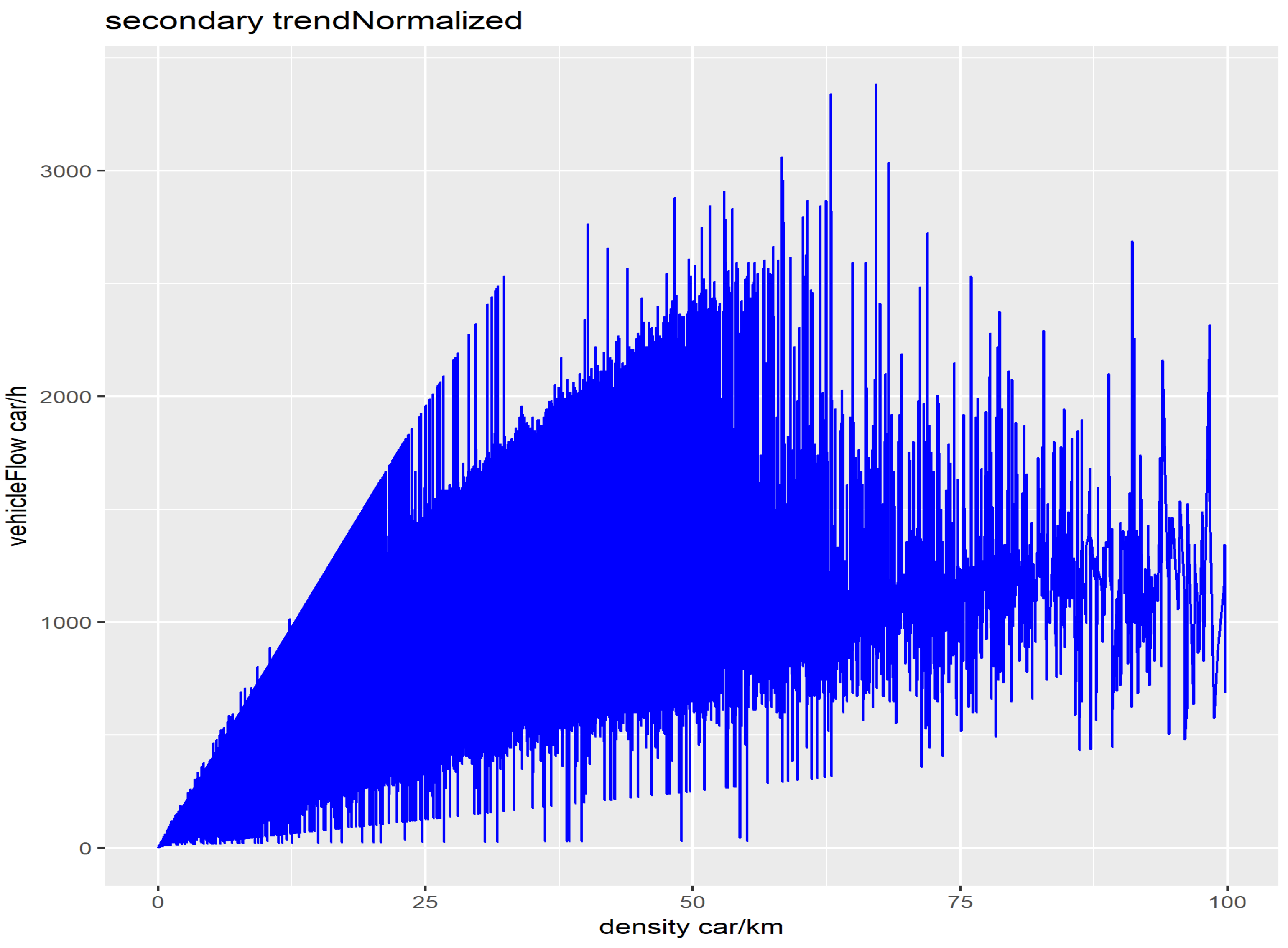
$$f(\rho) = v_{max}\left(1 - \frac{\rho}{\rho_{max}}\right)\rho$$

where v_{max} is the limit speed





Fundamental diagram: data analysis



Discretization scheme: at finite differences

On discrete domain the model is operated:

- Each road is partitioned in segments Δ*x* long.
- The time is partitioned in intervals Δ*t* long.
- Denote (h,m) a bounded time-space region (cell) of **duration** $\boldsymbol{h}$ and **length** $\boldsymbol{m}$. Let $u_m^h = u(t_h, x_m) = u(h\Delta t, m\Delta x)$ be a continuous function defined on (h,m). Denote *F* the numerical flux.
- Then, the vehicular density u_m^{h+1} results from:

$$\frac{\partial\rho(t,x)}{\partial t} + \frac{\partial f\big(\rho(t,x)\big)}{\partial x} = 0$$

$$u_m^{h+1} = u_m^h - \frac{\Delta t}{\Delta x}\Big(F\big(u_m^h, u_{m+1}^h\big) - F\big(u_{m-1}^h, u_m^h\big)\Big)$$

Sensors' measures

- The **measured data** sensor is interpreted as the **source of traffic** leading into the outcoming roads of the considered junction.
- **Boundary conditions:** assign a condition at the incoming boundary for $x = 0$ as $\rho(t,0) = \rho_b^{inc}(t)$.
- we proceed by inserting an "incoming ghost cell" and the discretization becomes

$$u_0^{h+1} = u_0^h - \frac{\Delta t}{\Delta x}\left(F\left(u_0^h, u_1^h\right) - F\left(v_{(inc)}^h, u_0^h\right)\right)$$

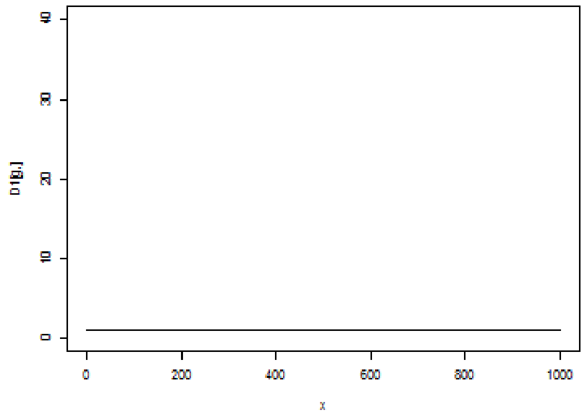
where

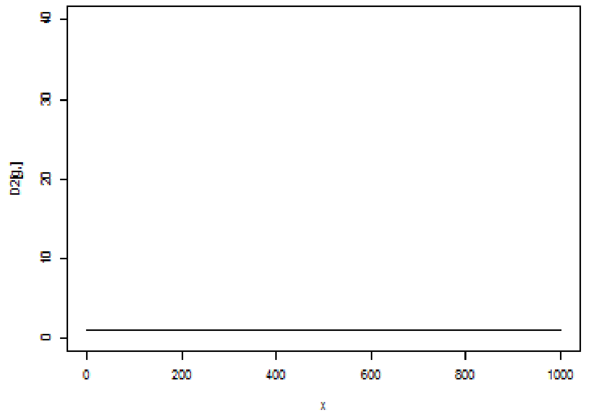
$$v_{(inc)}^h = \frac{1}{\Delta t}\int_{t_h}^{t_h+1} \rho_b^{inc}(t)\,dt$$

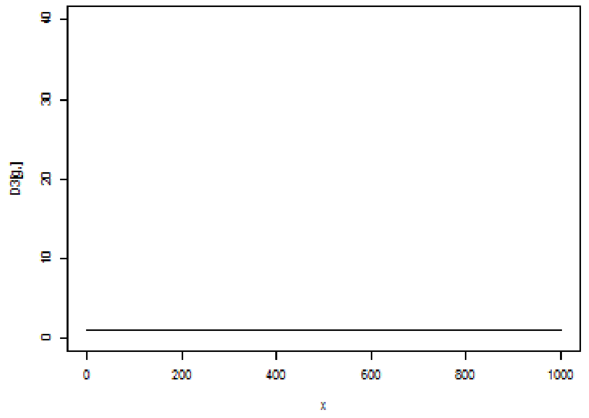
- replaces the "ghost value" u_{-1}^h

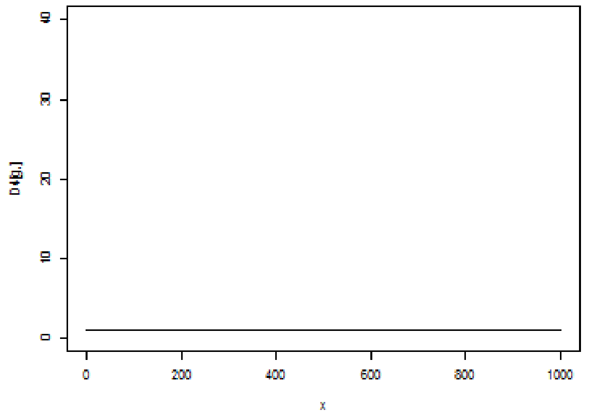
Fluid Dynamics Approach

Phisical Principle of Narrowing in roads









Application of the model

- For each time slot *t*, each traffic sensor detection is interpreted as a **source of traffic** that leads into the segments of road that origin from the node where the sensor is located that has produced the data.
- The vehicular traffic flow is propagated in the network according to the fluid dynamic model in (1).
- The distribution of the traffic at crossroads is governed by a **Traffic Distribution Matrix** whose coefficients are based on the ***weights*** of the segments of roads that make the crossroad.

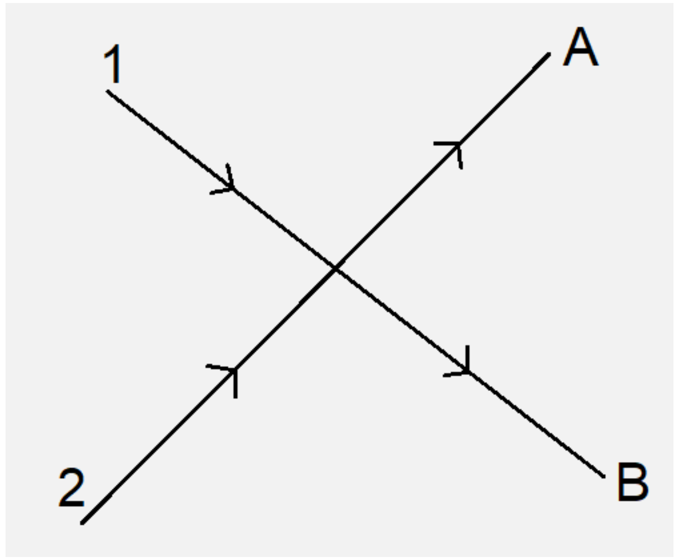
Traffic distribution at junctions

TDM are different for:

- each time slot over the day, draftly every hour,
- day kind: ferial, prefestive, and festive, etc.
- Junction geometry and semaphores settings, …

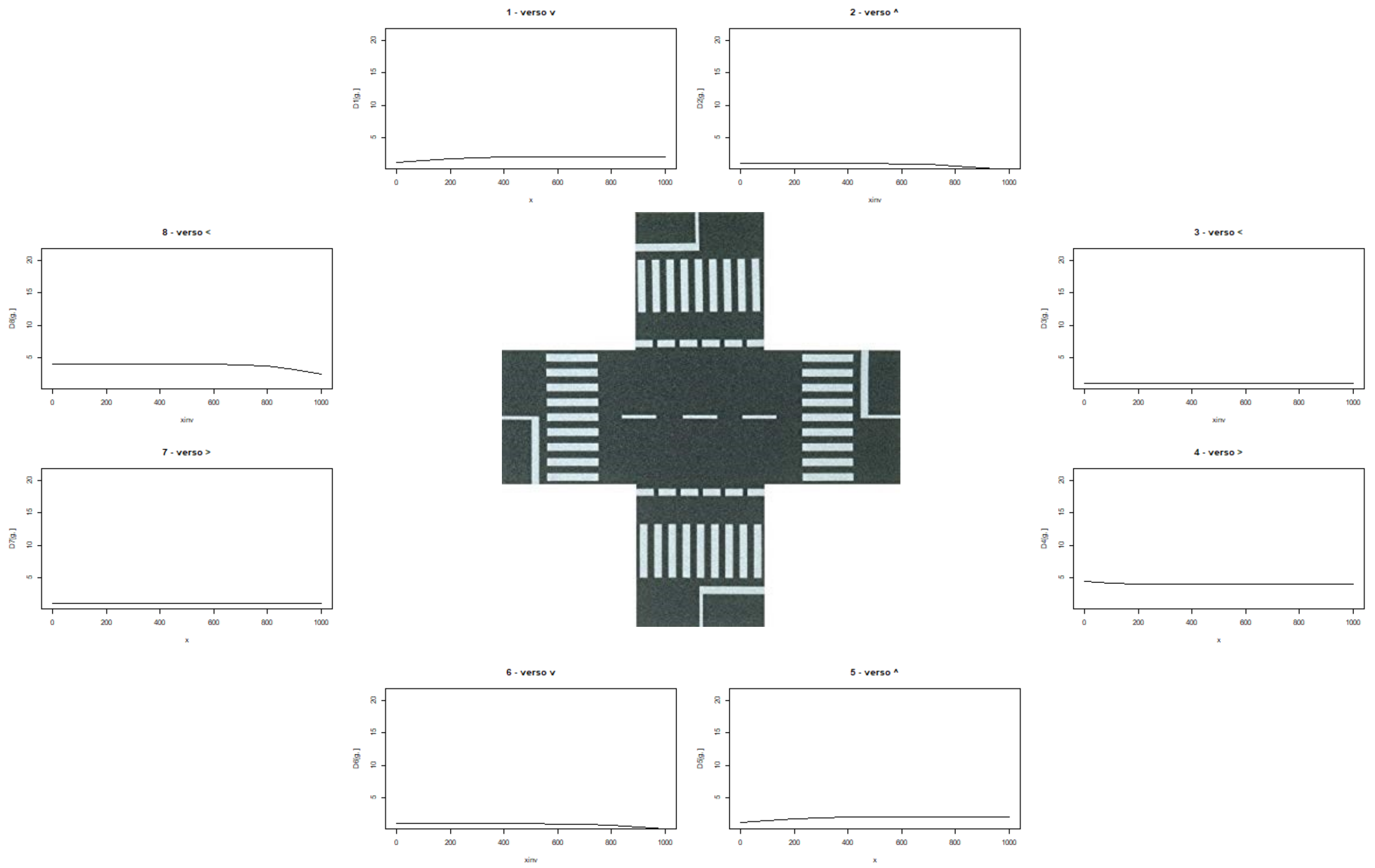
TDM can be:

- **Computed** draftly and ideally from the kind of roads: primary, secondary, etc.; number of lanes, etc.
- **Measured** by sensors at the cross roads (usually performed for tuning cross road lights, semaphores)
- **Learnt** from the measured traffic flows and from TFR

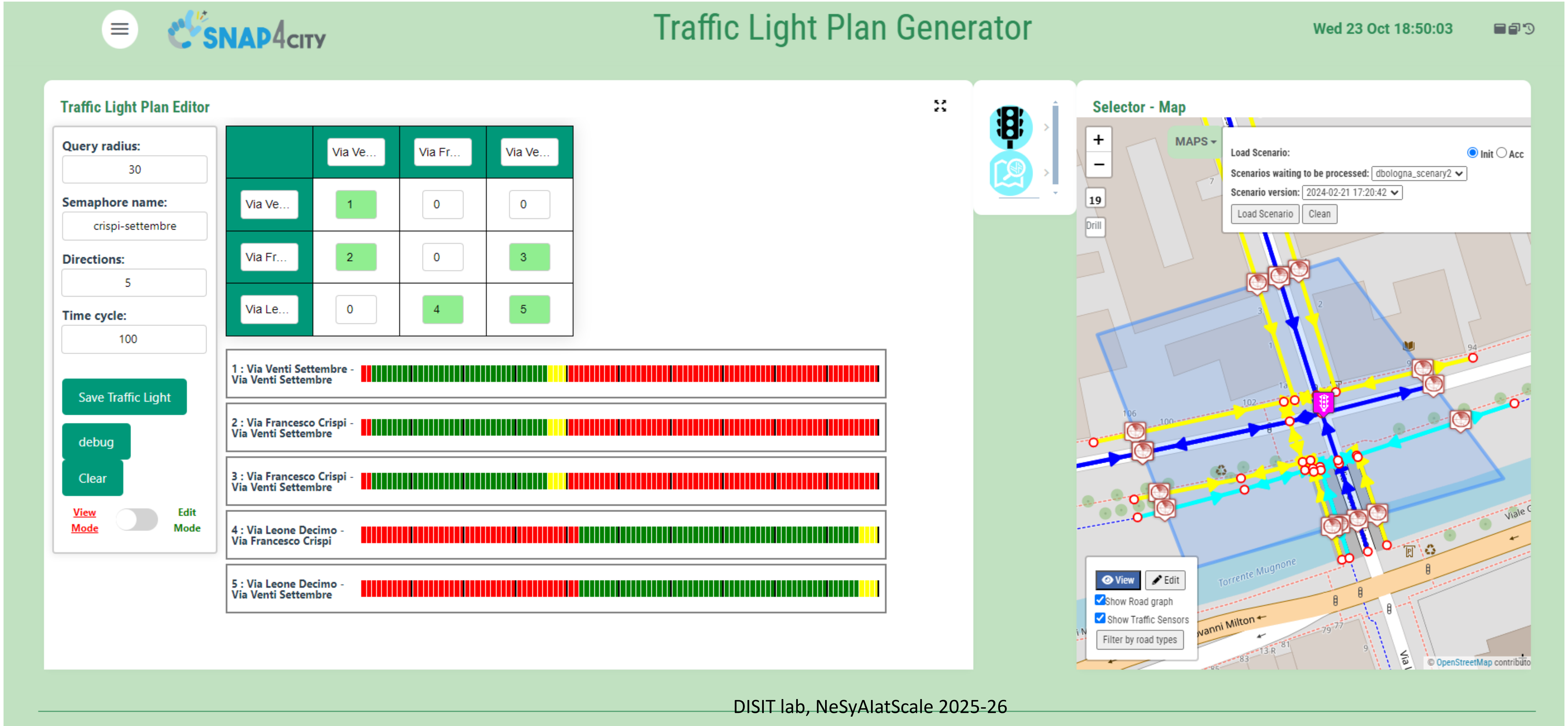


$$\begin{pmatrix} d_{1\mathrm{A}} & d_{2\mathrm{A}} \\ d_{1\mathrm{B}} & d_{2\mathrm{B}} \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = \begin{pmatrix} f_\mathrm{A} \\ f_\mathrm{B} \end{pmatrix}$$

Fluid Dynamics of a 4x4 Junction



Traffic Light Plan Editor



Traffic Light Plan Generator

Wed 23 Oct 18:50:03

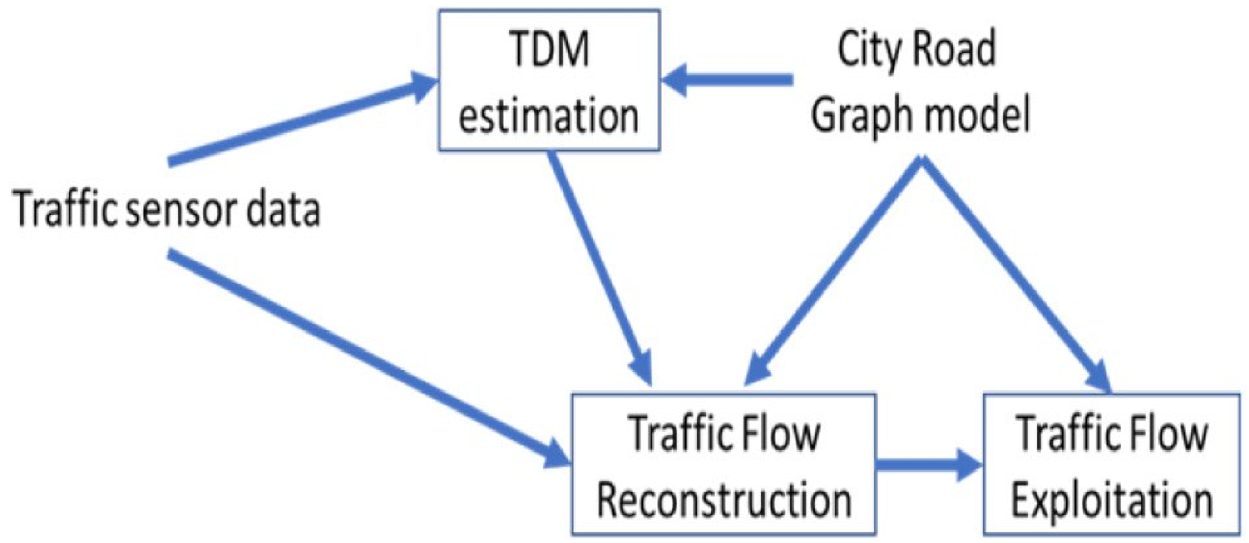
Traffic Light Plan Editor

Query radius: 30
Semaphore name: crispi-settembre
Directions: 5
Time cycle: 100

	Via Ve...	Via Fr...	Via Ve...
Via Ve...	1	0	0
Via Fr...	2	0	3
Via Le...	0	4	5

1 : Via Venti Settembre - Via Venti Settembre
2 : Via Francesco Crispi - Via Venti Settembre
3 : Via Francesco Crispi - Via Venti Settembre
4 : Via Leone Decimo - Via Francesco Crispi
5 : Via Leone Decimo - Via Venti Settembre

General Schema



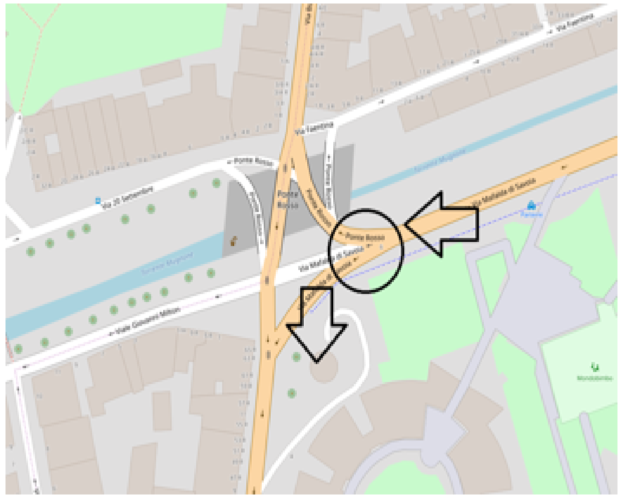
```
Load road network and details.
For each Time t:
        Get traffic sensors' values
        Load TDM(t) and weights
        For h: 1 → H
                Compute the model for each sensor' location to the
                sensors' value
                Compute the model for all traffic distribution in each
                junction
                Compute the model for all traffic density in each road-
                segments
        End For h
        Compute graphics representation
End For each Time
```

Junction distribution learning

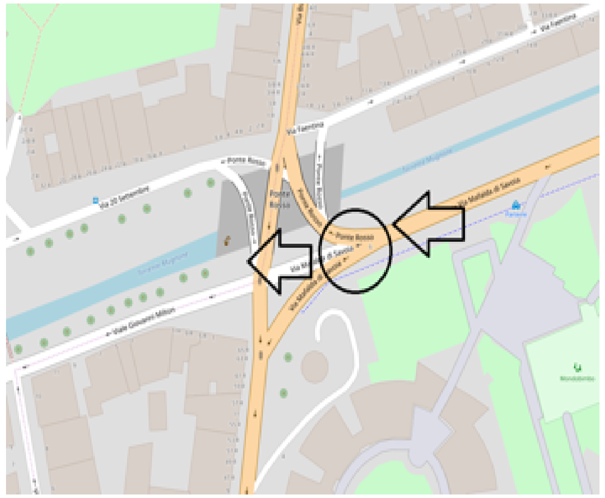
The fork of via Mafalda di Savoia (East), in via Mafalda di Savoia (South), Viale Giovanni Milton (West) and Via del Ponte Rosso (North), in Florence.

Junction distribution learning

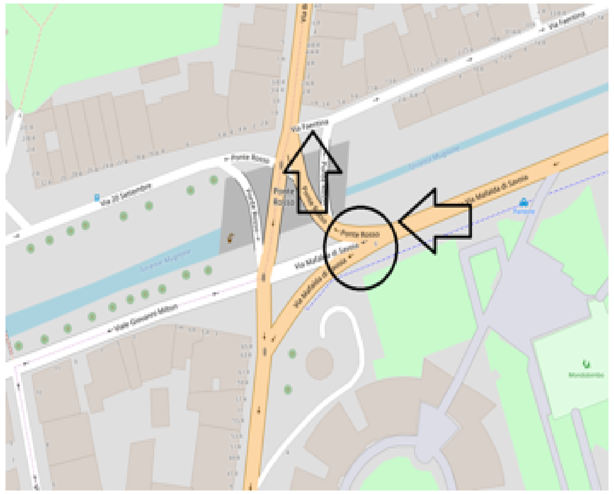
```
Road Type: primary
Lanes: 2
Designated Lanes: 0
Restrictions: none
Learning Factor: 61
Elem. Type: T.O.C.
Length: 63
Direction: positive
…
Weight: 31.122%
```



```
Road Type: tertiary
Lanes: 1
Designated Lanes: 0
Restrictions: none
Learning Factor: 24
Elem. Type: T.O.C.
Length: 51
Direction: positive
…
Weight: 12.245%
```



```
Road Type: primary
Lanes: 2
Designated Lanes: 0
Restrictions: none
Learning Factor: 111
Elem. Type: T.O.C.
Length: 60
Direction: positive
…
Weight: 56.633%
```



Weight initialization

Weights are **initialized** based on the following:

- **Road type**: motorway, trunk, primary, secondary, tertiary, unclassified, residential, service;
- **Lanes**: how many lanes are drawn on the asphalt, also considering possible restrictions (e.g. lanes reserved to public transport);
- **Traffic restrictions**: examples are mandatory/forbidden directions at crossroads, speed limits, limited traffic zones.

Stochastic relaxation learning

It has been observed that:

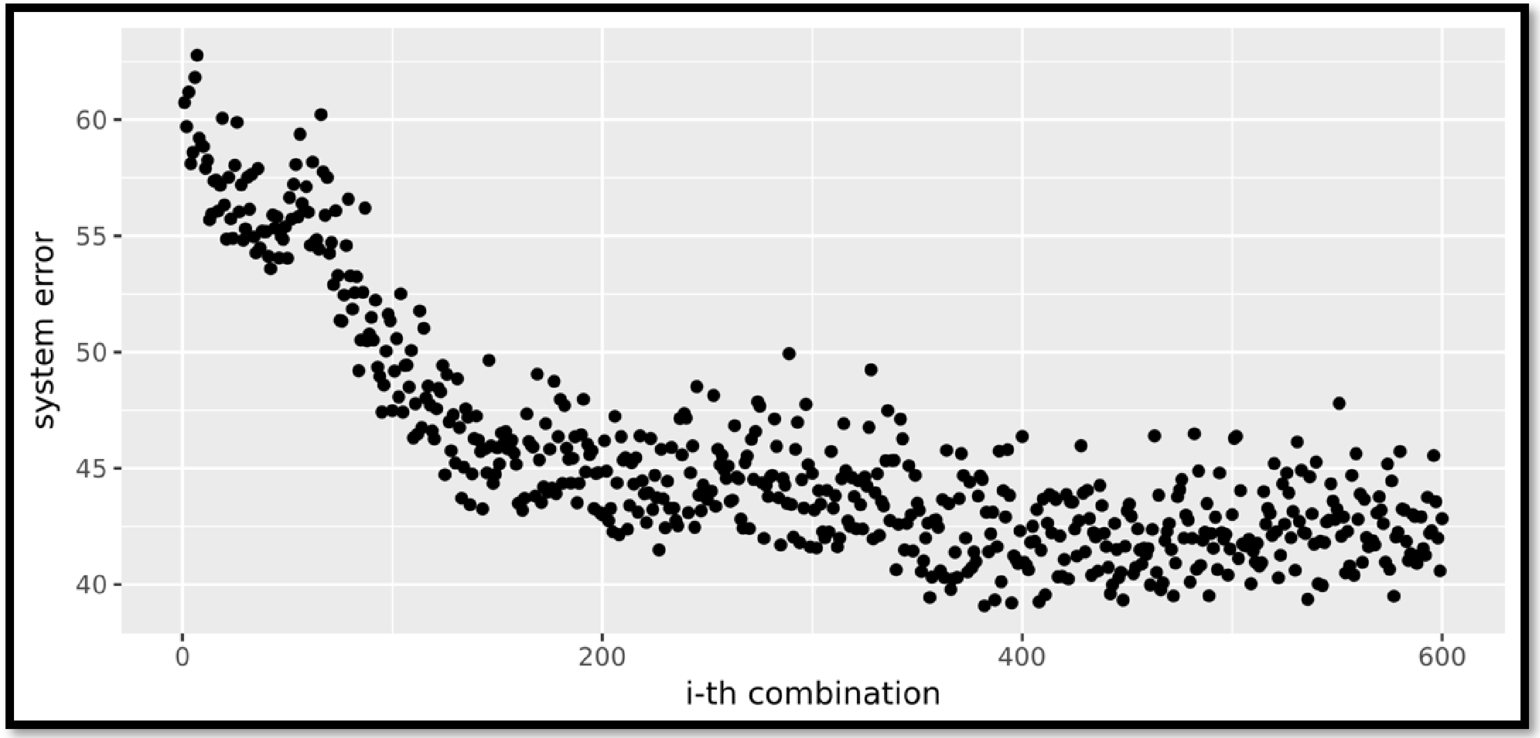
- The way how vehicles distribute at crossroads varies depending of the day of the week, and of the time of the day;
- A random variation of some weights is very likely to lead to an improved accuracy;
- If no improvements are achieved after *n* attempts, it is reasonable to move anyway to the best of the last *n* configs.

An offline process is run, based on the above, that leads to time-based weight adjustments, aimed at an improved accuracy.

Stochastic relaxation learning TDM

Simulated Annealing:

By minimising the differences on points of control via LOOCV as RMSE



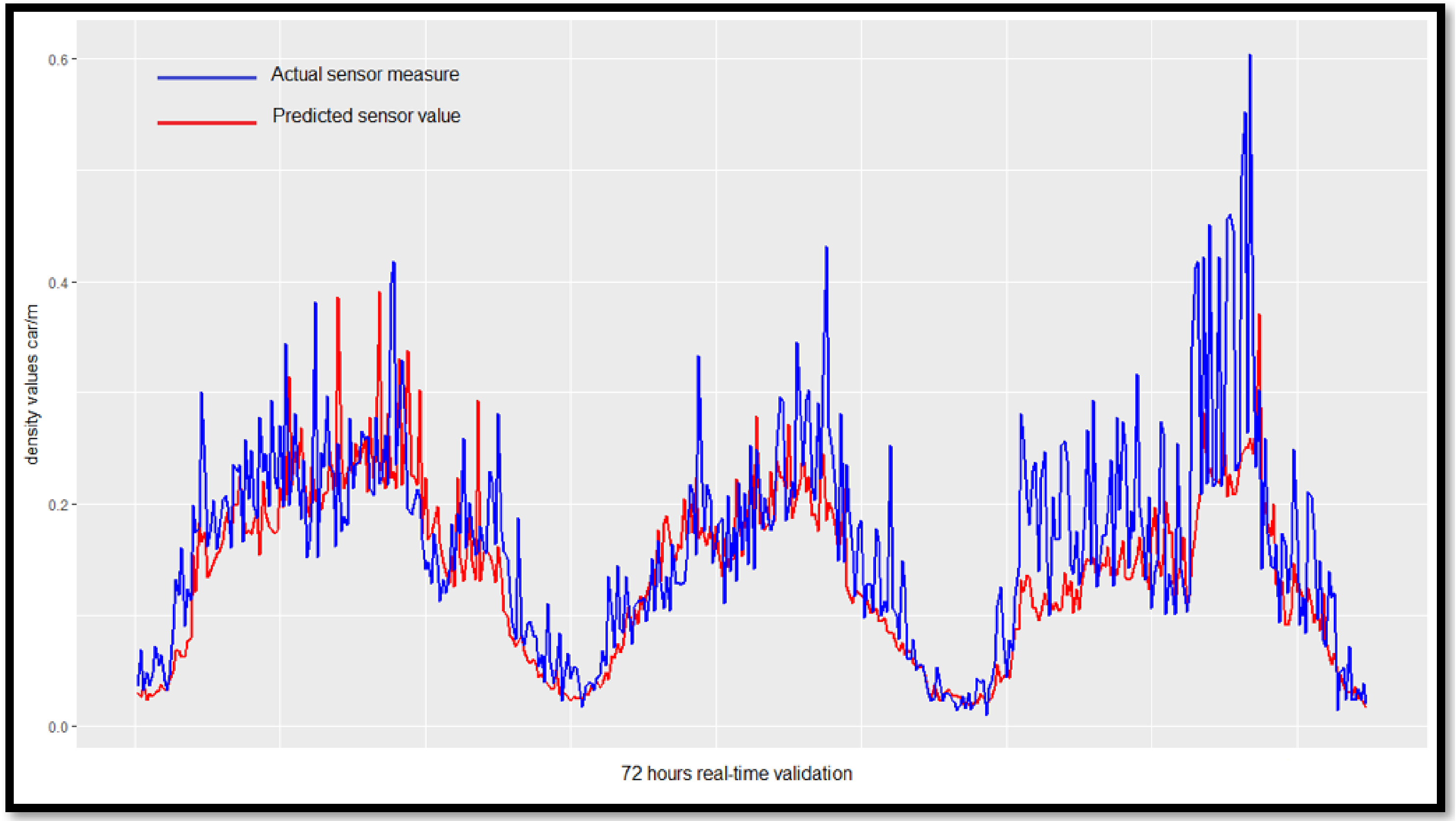
In the x axis, the number of the learning iterations. In the y axis, the (decreasing) system error.

Validation approach

- Once Obtained the TDM
- Let the **error at a sensor at a given time *t*** be the percentage error computed removing a *given sensor* from the inputs and comparing the traffic flow *reconstructed* at the sensor with the traffic flow *detected* by the sensor, at the given time
  - ***Leave-One-Out Cross Validation, LOOCV***.
- Let the **system error over a time period *T*** be the average of the system errors computed over all the traffic sensors and all the times ***t*** ∈ ***T***.

The system error has been computed to be the **30%** about.

Validation approach



The diagram refers to one in particular of the sensors, and it displays the predicted vs actual values over the time in the 72 hours validation.

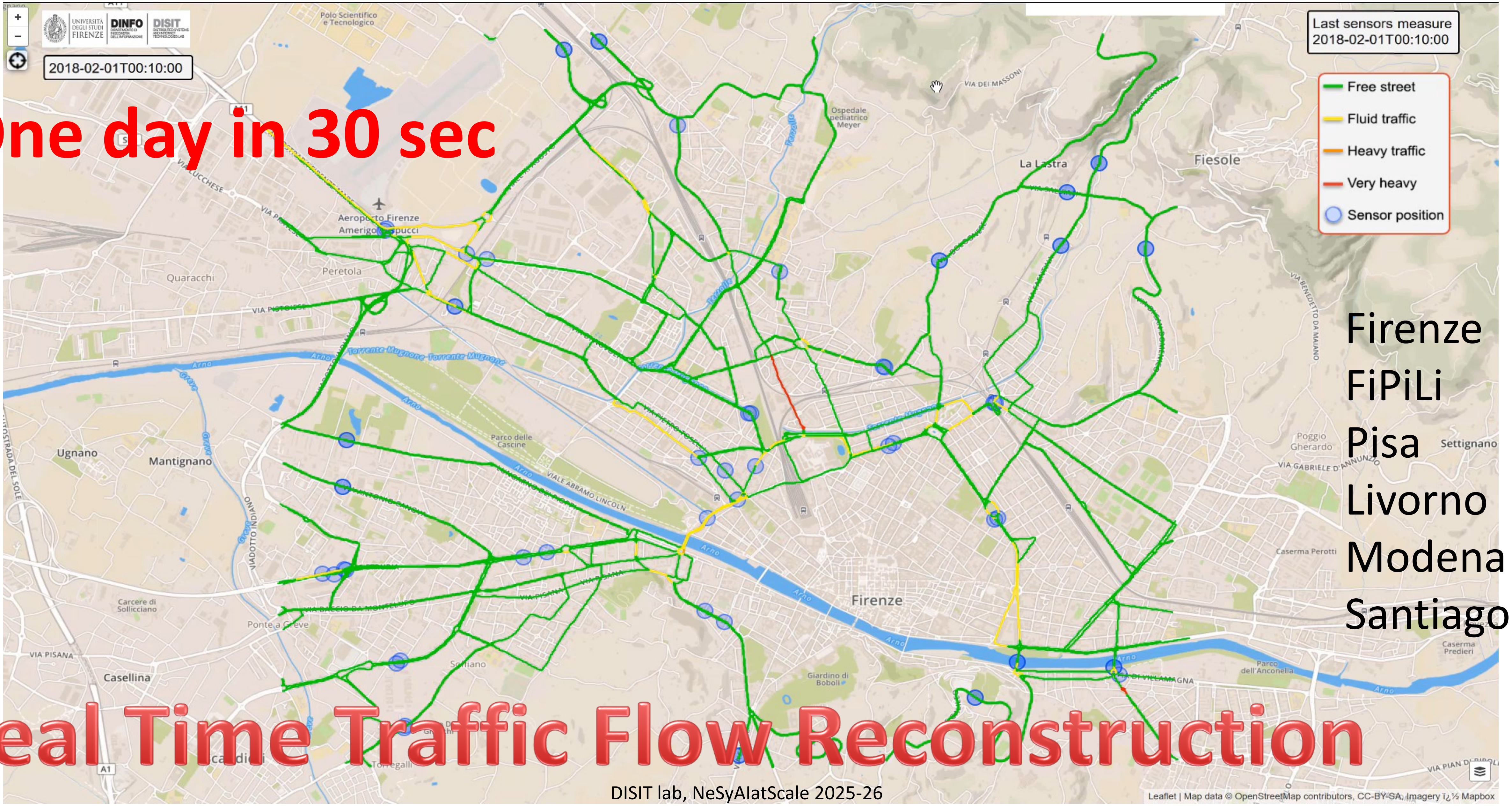
Displaying results

- Segments of road are categorized based on the road type and the number of lanes.
- Segments of each category that have one at least of the extremities that coincide with a traffic sensor, are used for determining the range of the traffic flows that can be observed on the specific category of segments.
- For each segment category, the range is partitioned into four subranges, that correspond to the four colors that you can find on the map.
- The reconstruction is presented to users through colored lines traced over the road paths on the city map.
- The date and time when the most up-to-date values from the sensors have been acquired can also be seen at the top-right corner of the map.

Displaying results

- **Road Segments** are categorized based on the road type and the number of lanes.
- Segments of each category that have one at least of the extremities that coincide with a traffic sensor, are used for determining the range of the traffic flows that can be observed on the specific category of segments.
- For showing
  - The reconstruction is presented to users through colored lines traced over the road paths on the city map.
  - For each segment category, the range is partitioned into four subranges, that correspond to the four colors that you can find on the map.
- The date and time when the most up-to-date values from the sensors have been acquired can also be seen at the top-right corner of the map.

Displaying results

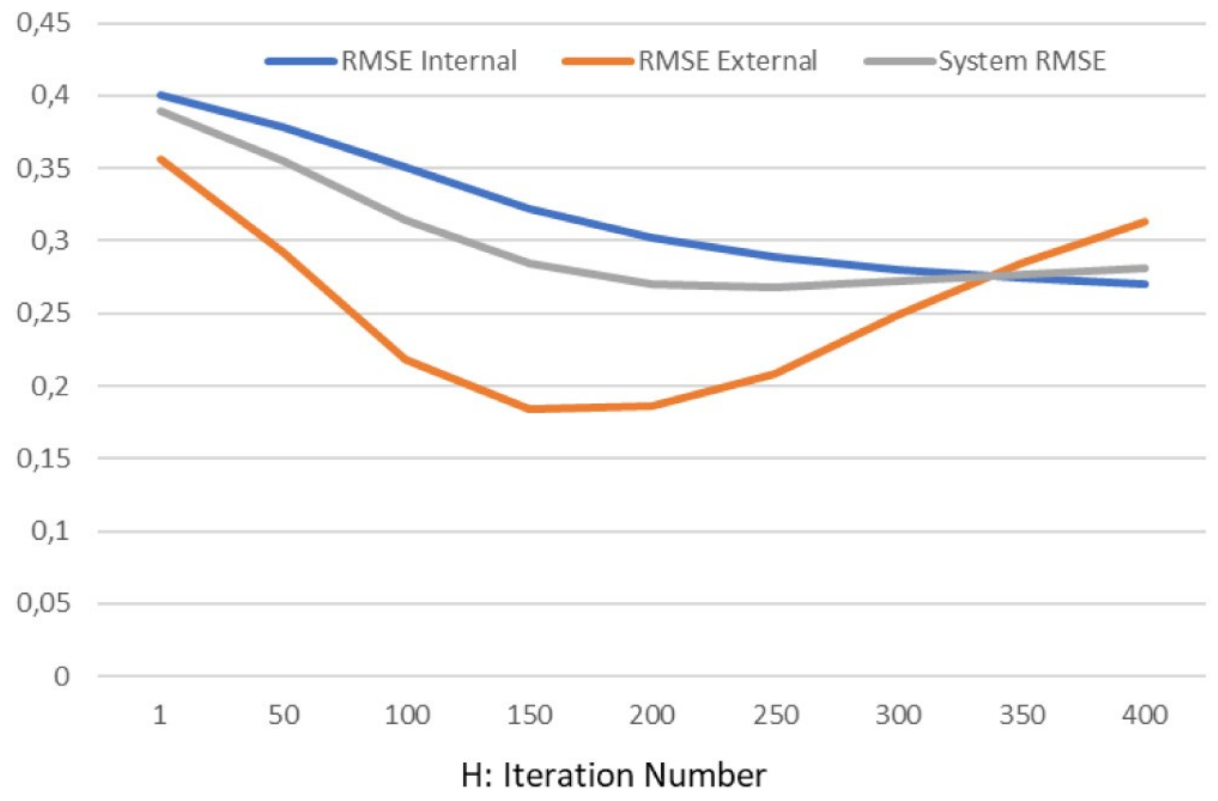


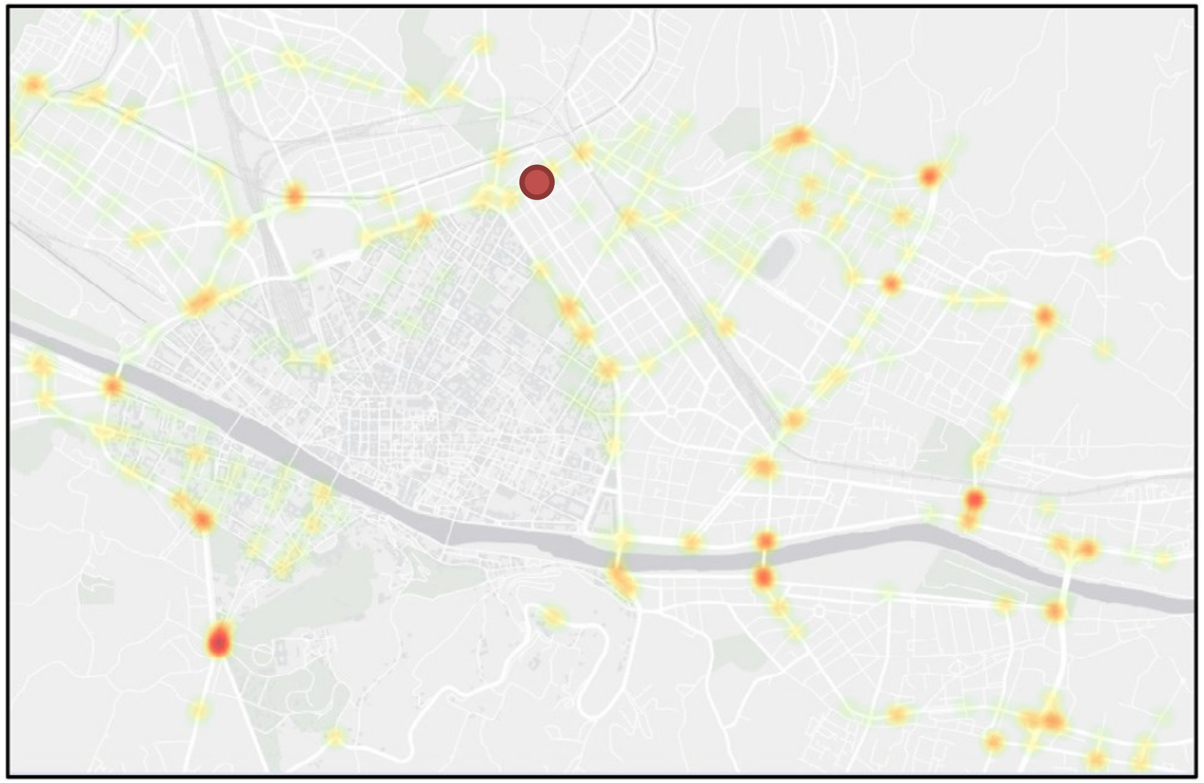
One day in 30 sec

Firenze
FiPiLi
Pisa
Livorno
Modena
Santiago

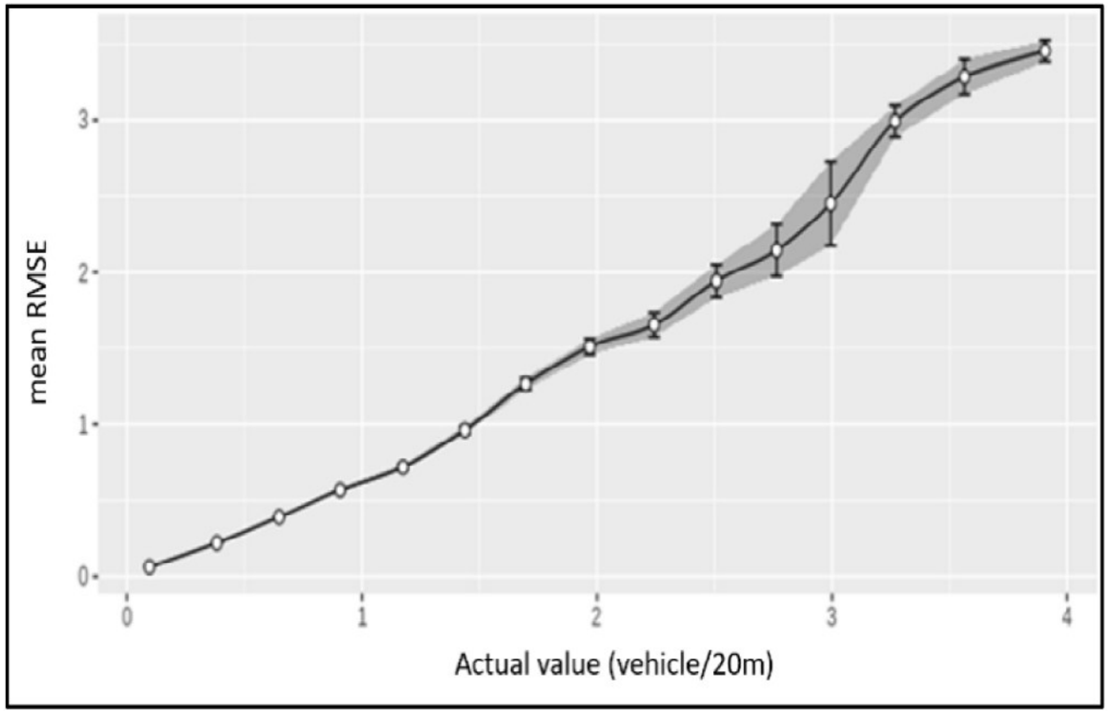
Real Time Traffic Flow Reconstruction

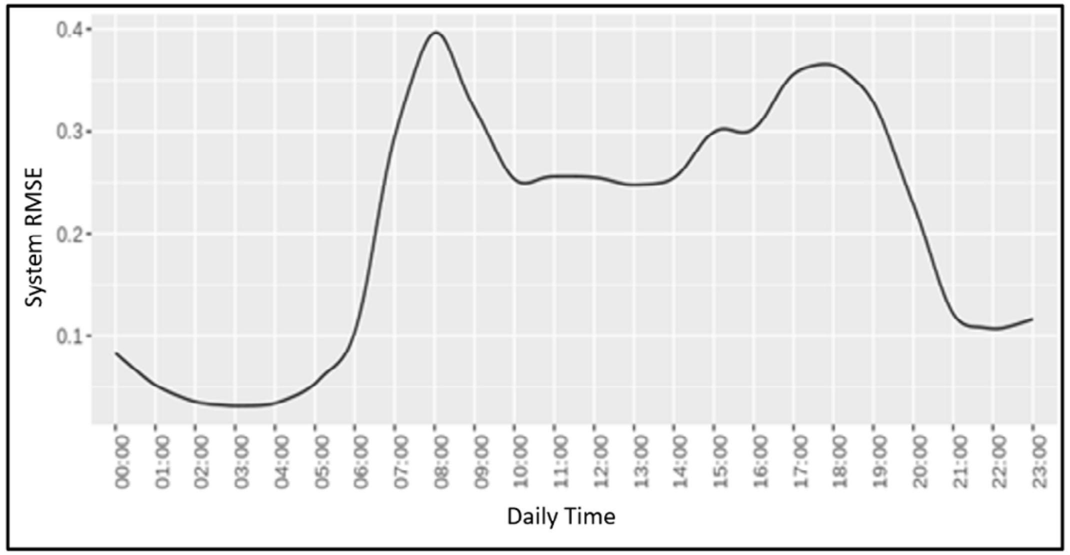
Error Assessment



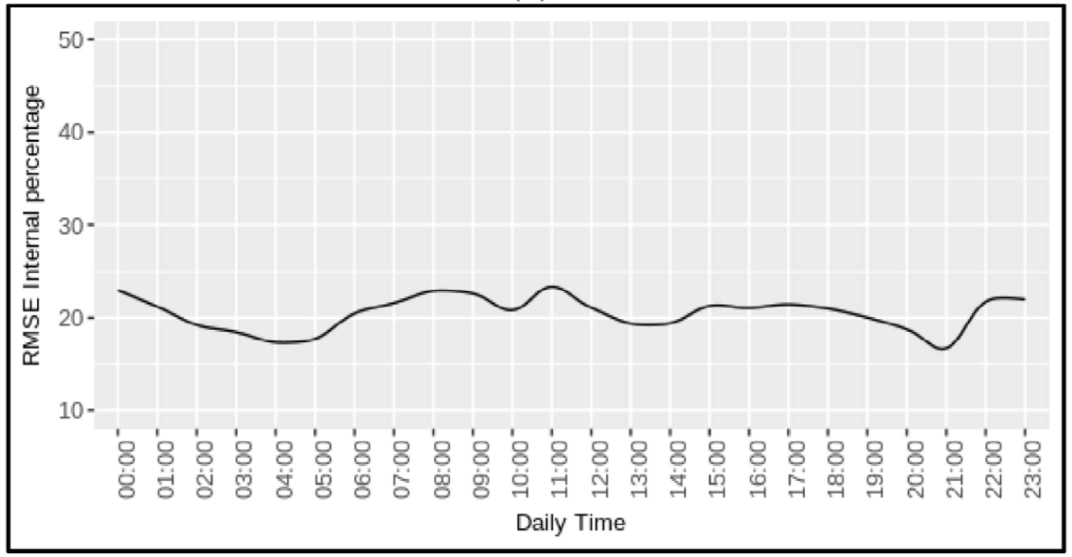

Betweeness

Error Assessment





(a)



(b)

ScienceDirect

- Stefano Bilotta, Paolo Nesi,
- **Traffic flow reconstruction by solving indeterminacy on traffic distribution at junctions,** Future Generation Computer Systems, Volume 114, 2021, Pages 649-660, ISSN 0167-739X, https://doi.org/10.1016/j.future.2020.08.017.

https://www.sciencedirect.com/science/article/pii/S0167739X20308359

10.2	6.125
CiteScore	Impact Factor

An example of Hybrid Approach

- **hybrid / neuro-symbolic (physics-based + neural)** method
  - – FEM (PDE solution at FD) + NN refinement of results

- **Finite Elements (FEM)** is physics-based, model-driven computation: it uses an explicit PDE + variational form + mesh/shape functions + solver.
  - – it is explicit structured knowledge (equations + discretization)
- **The NN refinement is sub-symbolic / numerical**
  - – learned continuous representation.

UNIVERSITÀ DEGLI STUDI FIRENZE

DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB

- S. Bilotta, V. Bonsignori and P. Nesi, "High Precision Traffic Flow Reconstruction via Hybrid Method," in ***IEEE Transactions on Intelligent Transportation Systems***, vol. 25, no. 5, pp. 4066-4076, May 2024, doi: 10.1109/TITS.2023.3329544.

https://ieeexplore.ieee.org/document/10315063

Use a PDE-based reconstructor to generate dense traffic states, then train ML models to reproduce (and improve) reconstructions faster, handling missing data and adding temporal features.

4066 IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 25, NO. 5, MAY 2024

High Precision Traffic Flow Reconstruction via Hybrid Method

Stefano Bilotta, Valerio Bonsignori, and Paolo Nesi, *Member, IEEE*

***Abstract*—Traffic management and sustainable mobility are the central topics for intelligent transportation systems (ITS). By means of modern technologies, it is possible to collect real-time traffic flow data to extract useful information to monitor and control vehicular traffic. On the other hand, costs to obtain this piece of information are high. It requires either direct measures in the network road by installing large number of sensors (more precise data) or acquiring data from international providers supplying data coming from onboard units, mobile app, navigators, etc. In current paper, this problem has been addressed providing a solution granting traffic flow data in each road segment of the whole network by reconstructing the computation by means of data from few scattered traffic sensors in fixed positions of the road network. The proposed approach combines the solution of nonlinear Partial Differential Equations (PDEs) with machine learning for improving the state-of-the-art solutions of PDE. The result has been a higher precision with respect to PDE-based solutions, and a strongly reduced execution time. Several different machine learning models have been compared for such a purpose, demonstrating the general viability of the hybrid architecture proposed. The research result has been obtained in the framework of both the Sii-Mobility national project on transport systems, and MOST, the National Center on Sustainable mobility (both funded by the Italian Ministry of Research), by exploiting the Snap4City platform.**

***Index Terms*—Traffic flow reconstruction, traffic flows, machine learning, hybrid architectures, machine learning PDE solution.**

I. Introduction

TRAFFIC flow computation consists in obtaining real time traffic flow state in each segment of a road network in a urban or rural area. Such a computation is fundamental for implementing a large number of smart services such as: dynamic route guidance, road digital signage, congestion detection, traffic reduction; fuel consumption and pollution emission monitoring, etc. [1], [2]. Often, traffic flow estimation is related to a monitored area based on few fixed points/sensors and thus no information is provided in other connected road segments free of sensors. Many contributions focus on this field of research as in [3], [4], [5], [6], [7], [8], and [9]. The usage of large number of traffic flow sensors can help in getting more precise estimations in the whole city (road network), but costs may become unaffordable. Traffic density measures are typically obtained by stationary sensors on fixed positions and they are usually of different kinds: TV cameras, road spires, etc. [10], producing measures in terms of traffic flow density, velocity and number of vehicles. Due to sustainability reasons, the number of deployed sensors has to be limited. Thus, it is mandatory to adopt some reconstruction algorithms to obtain the traffic flow condition in each road segment of the city in order to have dense traffic flows in the unmeasured road segments.

Surrogated traffic flow data can be obtained from: Mobile Apps, on board units (insurance black boxes for instance), social media app [11], and recently also from vehicle networks [12]. In [13], a deep Restricted Boltzmann Machine and Recurrent Neural Network, RNN, architecture has been used to predict traffic congestion evolution based on GPS data from taxis, and thus on their position and velocity, etc. In [14], a smartphone-based crowd sensing system for traffic detection and measure has been proposed, where data are gathered from the handheld devices. Data coming from navigator Apps (e.g., TomTom, Google map, Waze), at long term, could be very expensive for a municipality with respect to the installation of sensors. Those measures are not related to the actual counting of vehicles, since they are based on measuring single vehicle velocity, which does not directly relate to road traffic density. Vehicular Ad-Hoc Networks, VANET, are modeling communication among vehicles, thus creating a shared network of information which could be used to understand local traffic [12], [15]. On the contrary, the usage of TV Cameras located in specific critical points allows to perform direct measures, which reduces costs, while increasing precision in specific points. Then, multiple areas/lanes can be controlled with a single installation, so as to enable the control of a high number of traffic flows. Traffic flow sensors provide continuous measuring of traffic on selected roads at fine grain, and in most cases, they also provide information about the kinds of vehicles: busses, tracks, cars, bikes, etc. Generally speaking, to setup a network of traffic flow sensors in a city drastically avoids the costs of taking updated data from third parties such as Google or Navigator mobile Apps, which provide statistical data, instead of specific and direct measures.

A. Related Work

In the context of traffic flow theory, a distinction has to be done between traffic flow ***Predictions*** in specific points in urban contexts or highways (short or long terms in the

Manuscript received 19 December 2022; revised 26 July 2023 and 16 September 2023; accepted 30 October 2023. Date of publication 10 November 2023; date of current version 13 May 2024. This work was supported by the CN MOST, National Center on Sustainable Mobility. The Associate Editor for this article was S. Ahn. *(Corresponding author: Paolo Nesi.)*

The authors are with the DISIT Laboratory, DINFO Department, University of Florence, 50139 Florence, Italy (e-mail: stefano.bilotta@unifi.it; valerio.bonsignori@phd.unipi.it; paolo.nesi@unifi.it).

Digital Object Identifier 10.1109/TITS.2023.3329544

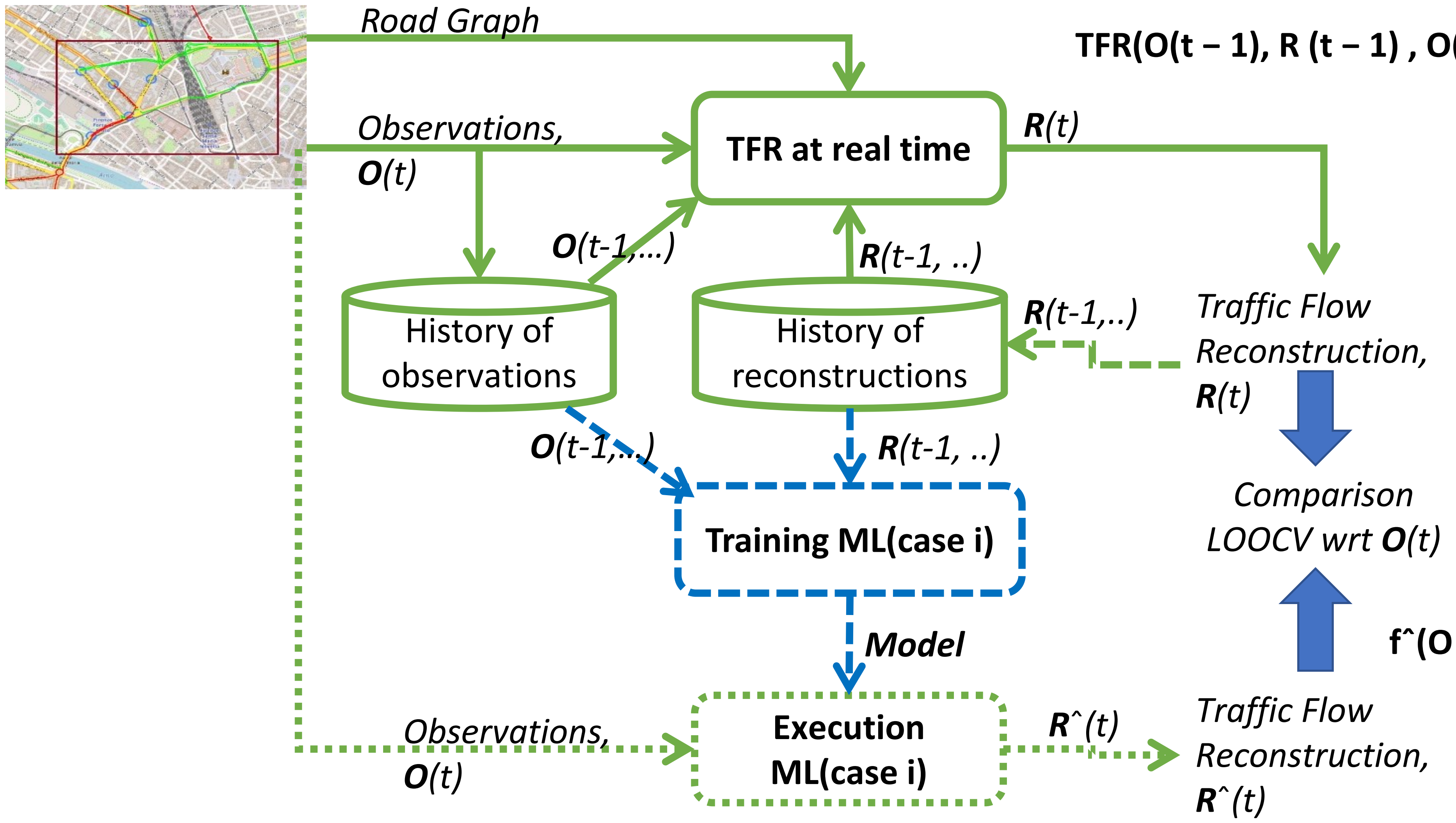
© 2023 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

High-Precision Traffic Flow Reconstruction via a Hybrid PDE + ML Method

- How it is possible to combines nonlinear PDE traffic modeling with ML to improve accuracy and estimate TFR at runtime
- **Traffic Flow Reconstruction, TFR**
  - Passing from scattered data to dense data on traffic flow solving PDE, + knowing data in a number of points, it is an ill-posed problems
  - **solving indeterminacy at junctions**, and adding other hyp/equations
    - At junctions, measured inflows do not uniquely determine outflows.
    - Distribution patterns (turn ratios) vary with time and context.

Hybrid Traffic Flow reconstruction, Case (i)

TFR(O(t – 1), R (t – 1) , O(t), RoadGraph) → R(t)



$$\Delta R(t) = \frac{1}{n}\sum_{z=1}^{n} |\hat{R}_z(t) - R_z(t)|$$

f^(O (t)) → R^(t)

Why to improve the TFR?

- **The above graph:**
  - **O(t) is** TF density are the observations from sensors.
  - **R(t) is** TFR estimated using DPE+LOOCV (200 iterations), assuming TDM estimated from a previous process.
- **Needs**
  - Iterative PDE + stochastic TDM estimation is expensive.
  - The hybrid method aims to keep physical consistency while reducing runtime and improving fit at sensor locations.

Hybrid idea

- **Core principle**: keep the model-driven reconstructor, but use it to produce training targets that an ML model can learn quickly, to refinement, and faster run.
- The PDE solver encodes traffic physics and network constraints.
  - ML learns a fast approximation of the expensive junction-distribution optimization — and may add patterns not captured by the PDE baseline

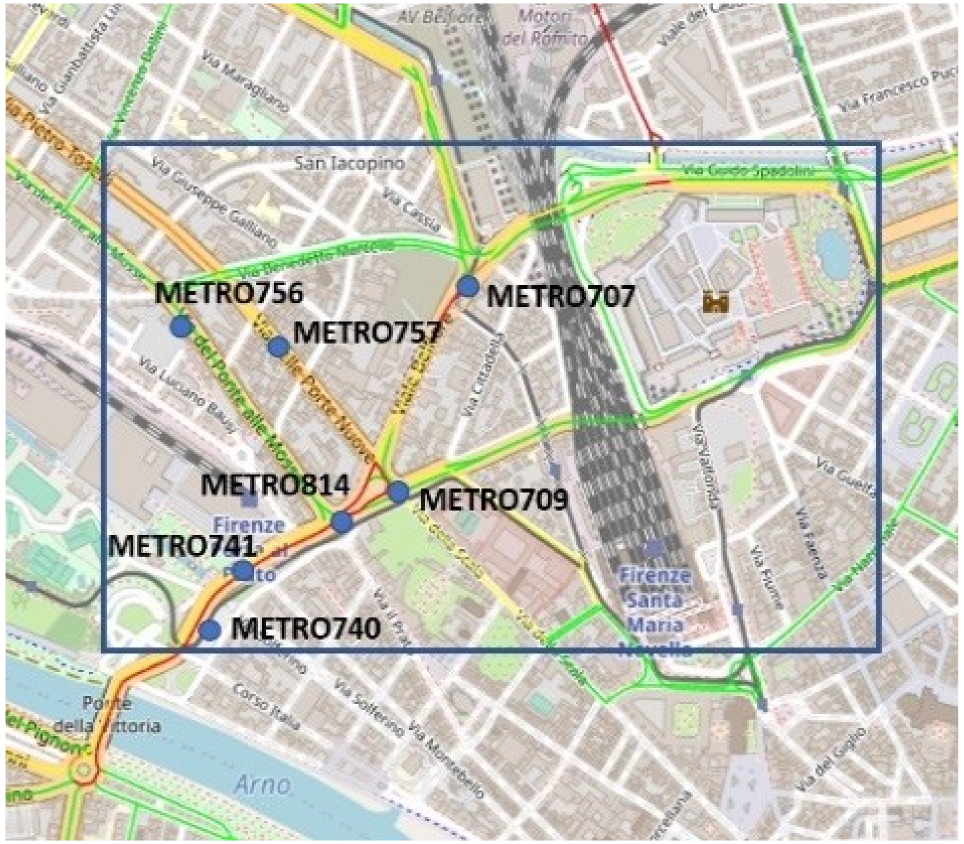
Learns da O(t)→R(t) per fare **O(t)→R^(t)** **f̂(O (t)) → R̂(t)**

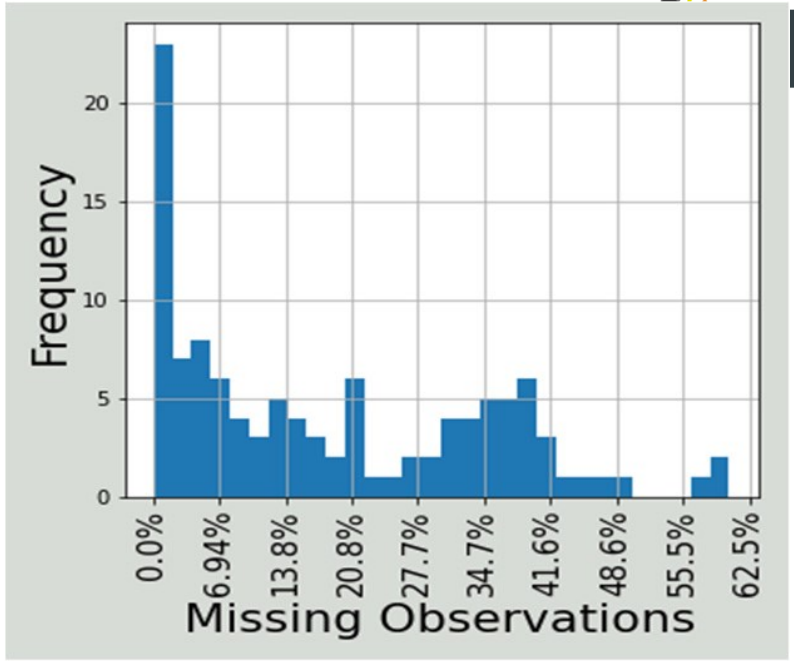
ML is trained without explicit temporal features from data of O()+R();
it learns a direct mapping from sensor observations to dense state TFR by PDE

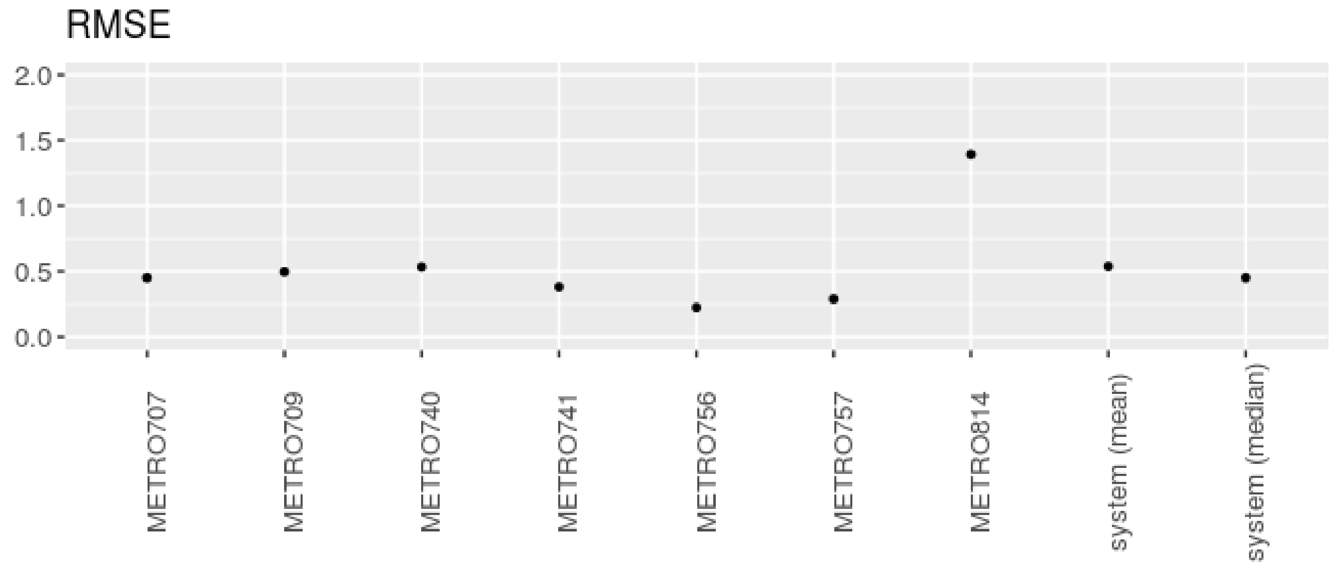
1) Improve accuracy vs PDE-only reconstruction
2) Reduce execution time at runtime
3) Keep performance when some sensor samples are missing
4) Scale to large road networks

UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB

TFR from PDE+TDM







Case value and Assessment

- **Subnet of**
  - 7 traffic sensors, O(t)
  - 103 intersections / junctions
  - ~735 road segments (20m units) in the subnet
  - Dense reconstruction computed for ~728 units
- **Training**: Nov 2019 → Feb 2020; data sampled roughly every ~10 minutes; missing samples occur due to faults/maintenance.
  - 13,208 observation timestamps
  - Ground truth is only directly observed at sensor locations.
  - LOOCV assesses reconstruction quality by excluding one sensor at a time.
- **ΔR compares ML dense reconstruction vs TFR** dense outputs to ensure no degradation elsewhere

$$\Delta R = \frac{1}{T}\sum_{t=1}^{T} \Delta R(t). \qquad \Delta R(t) = \frac{1}{n}\sum_{z=1}^{n} |\hat{R}_z(t) - R_z(t)|$$

Metriche

- Valutazione wrt un sensore:

$$RMSE(k) = \sqrt{\frac{\Sigma_{t=1}^{T} \left(R_k\,(t) - O_k\,(t)\right)^2}{T}},$$

$$MAE(k) = \frac{\Sigma_{t=1}^{T} \left(|R_k(t) - O_k(t)|\right)}{T}.$$

- Valutazione in tutti gli m sensori in LOOCV

$$RMSE(system) = \frac{1}{m} \sum_{k=1}^{m} RMSE(k).$$

Assessment

- **Leave-One-Out Cross Validation** (LOOCV): exclude sensor k, reconstruct its road segment, compare to its observation.
- **Units are vehicle density** (e.g., vehicles per 20m). Errors increase when traffic density is high (rush hours), but normalized error ratios are relatively stable (paper notes ~25%).
- **Only sensors have ground truth**, the paper also compares dense outputs: ΔR measures the average absolute difference between ML reconstruction $\hat{R}(t)$ and TFR reconstruction $R(t)$ over all reconstructed units.

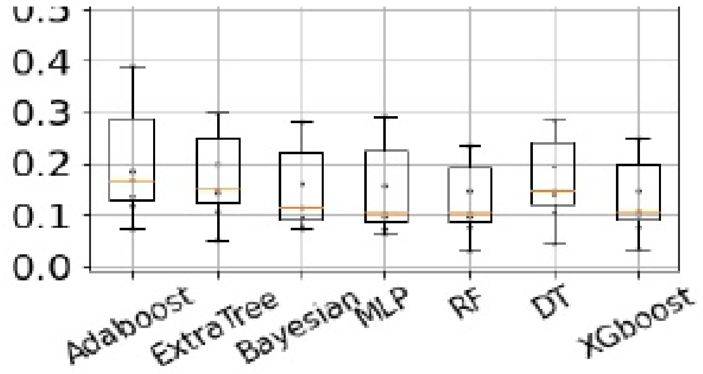
Using and comparing different ML regressors

- The paper compares multiple ML regressors to learn dense reconstruction from sparse sensor observations.
- **Ensembles:**
  - AdaBoost, adaptive boosting
  - Random Forest (RF)
  - XGBoost: gradient-boosted decision trees
  - Decision Tree (DT)
  - ExtraTrees
  - Bayesian regressor
- **Neural:**
  - MLP, Multilayer Perceptron (1 hidden layer)
    - **ReLU:** Rectified Linear Unit, ativation
    - **Adam**: Adaptive Moment Estimation, combines the advantages of Momentum and RMSprop techniques to adjust learning rates during

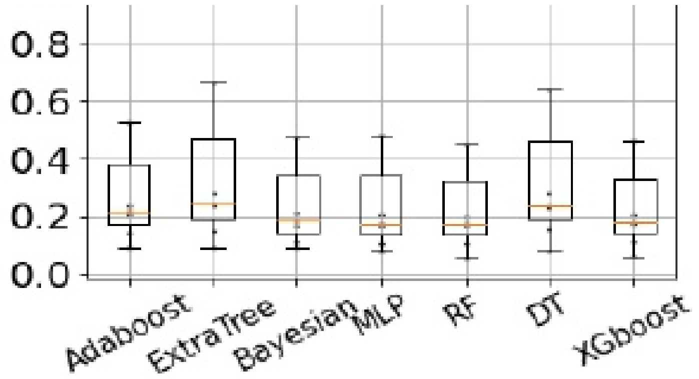
Tree ensembles can model nonlinear cross-sensor dependencies and handle feature interactions with relatively little feature engineering, which helps when only a few sensors are available.

assessment

- For PDE TFR, MAE(system) = 0.4, and RMSE(system) =0.53
- improvement has been in the range of delta MAE(system) of 0.22,
- a reduction of more than the 50% of the TFR PDE error in estimating traffic flow.



MAE(system)

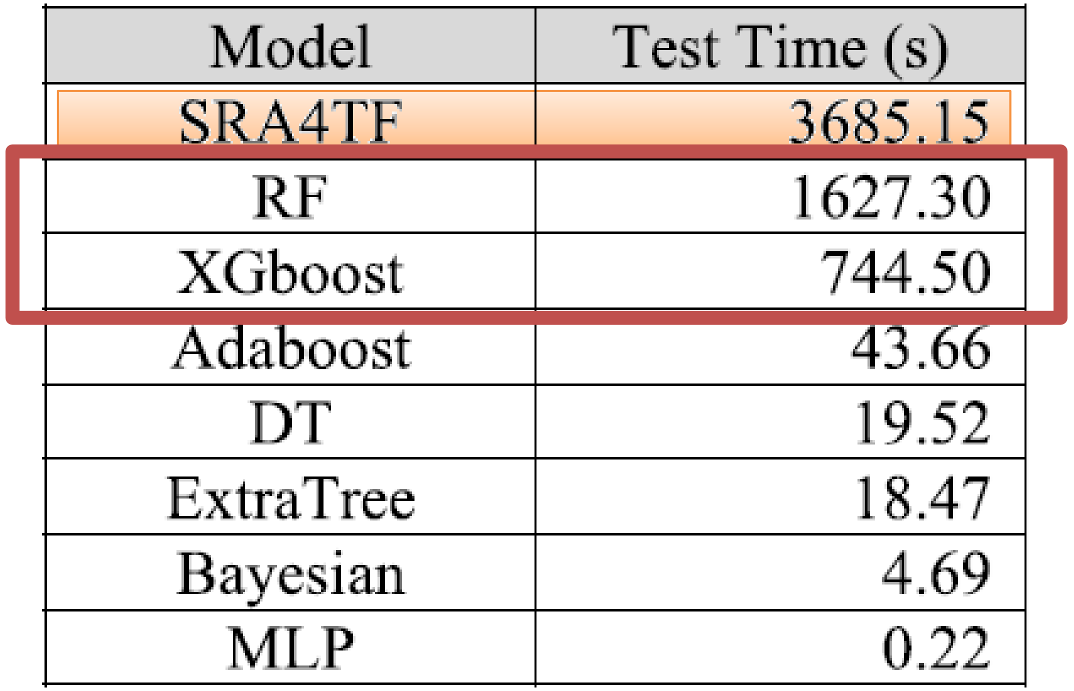


RMSE(system)

Model	*ΔR*
Bayesian	0.0942
Adaboost	0.0848
MLP	0.0676
ExtraTree	0.0552
DT	0.0519
XGboost	0.0467
RF	0.0435

Performance in execution

- executions have been conducted on a GPU board NVIDIA Quadro GV100 with 32GByte Ram, which has 5120 CUDA Cores, FP64 perf as 7.4 TFLOPS.
- MLP 16,000× speedup
- SRA4TF = TFR PDE



Model	Test Time (s)
SRA4TF	3685.15
RF	1627.30
XGboost	744.50
Adaboost	43.66
DT	19.52
ExtraTree	18.47
Bayesian	4.69
MLP	0.22

Deploy at scale

A practical trade-off: the best accuracy model may not be the fastest.

- Run TFR PDE TDM to periodically to refresh training data / calibrate
- Use ML at runtime for fast TFR estimation
- Monitor drift at sensors; retrain when needed

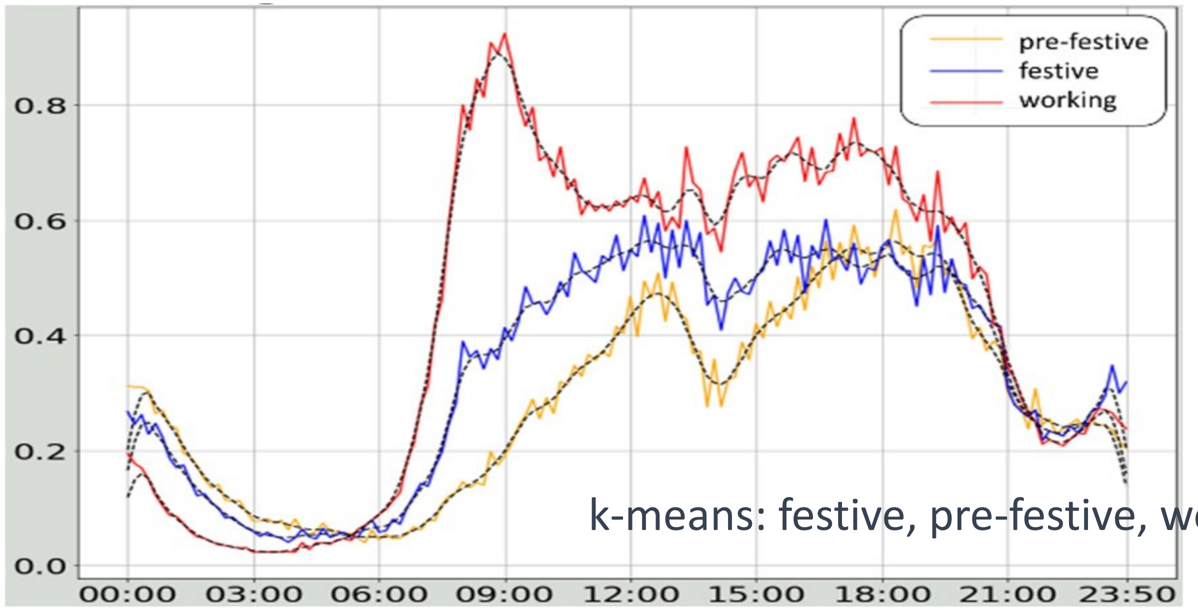
Use control: if ML predictions deviate too much at sensors, fall back to PDE-only, or re-run TFR PDF for TDM estimation update.

Case (ii)

Case(i): $\hat{f}(O(t)) \rightarrow R(t)$

Case(ii): $\hat{f}(O(t), h(t), d(t)) \rightarrow R(t)$

Real sensors drop samples.
When missing rate is high, PDE-only reconstructions may fail.
Case (ii) introduces features that can still be computed a priori (day type, time slot) and pairs with imputation strategies.



k-means: festive, pre-festive, working.

adding explicit temporal features and addressing discontinuities in sensor data

- h(t): time-of-day encoding (**hours** or coarse time slots)
- d(t): **day-type** / typical trend information (festive vs pre-festive vs working)
- *Rationale*:
  - traffic has strong daily/weekly seasonality;
  - temporal features help disambiguate patterns from sparse sensors.

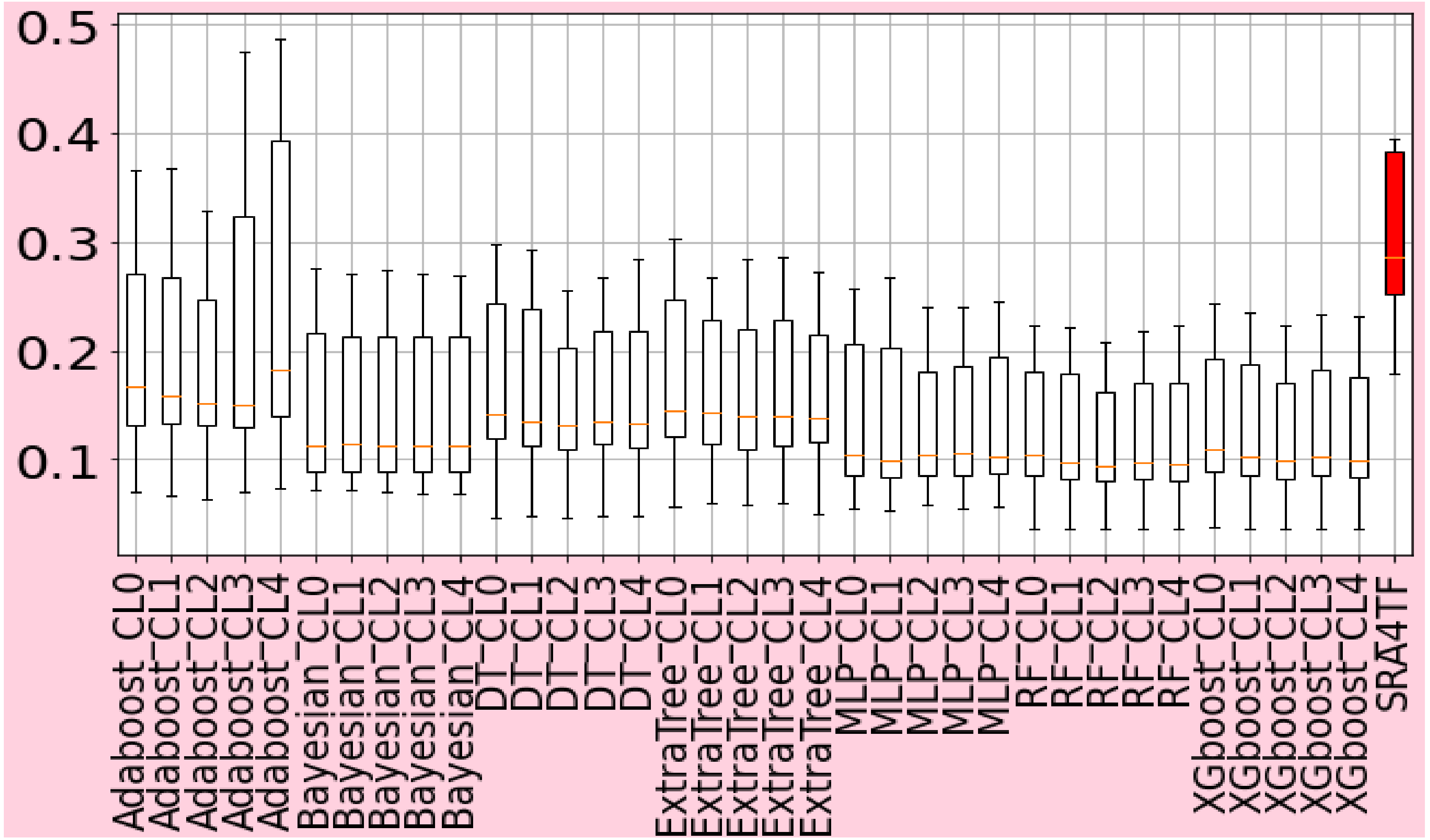
Un Case(i) and 4 Cases(ii)

Case (i) as CL0

Cases(ii)

- CL1 is the case where the d(t) are 3 classes and h(t) are in 4 time slots.
  - They are coded together into 3×4 = 12 possible values in a unique input data encoded together.
- CL2 is the case where d(t) are 3 classes, and h(t) are in 24 time slots.
  - They are separately encoded.
- CL3 is the case where d(t) are 3 classes, and h(t) are in 4 time slots,
  - while they are separately encoded, which makes it different from CL1.
- CL4 is the case where d(t) are 3 classes, and h(t) are in 48 time slots.
  - They are separately encoded.

Comparisong among different NN solutions

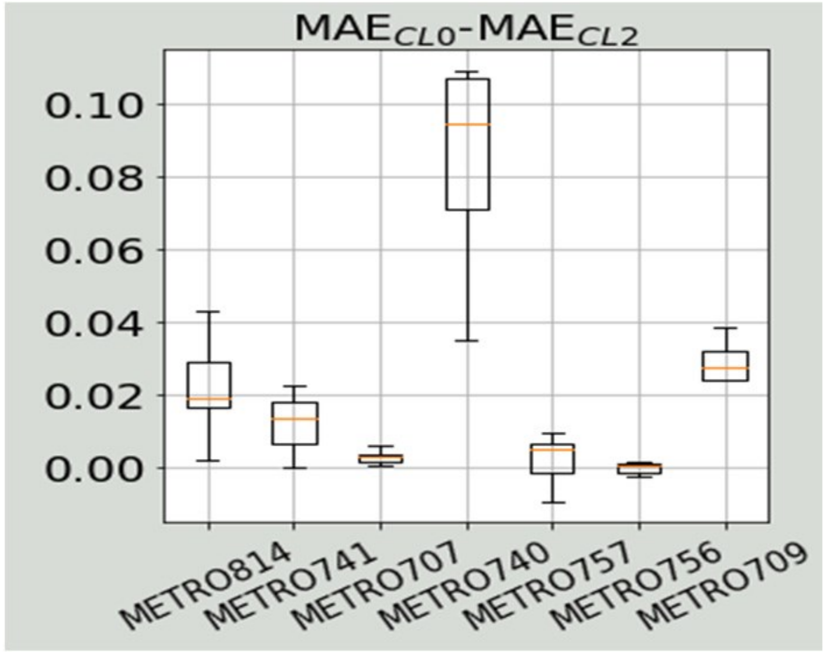


RF resulted the best in increase the precision of TFR in the network

Resulting MAE close to 0.1

Considerations

- Cases (ii) are generically better than Case(i), but not all.
  - CL2 **is the case where d(t) are 3 classes and h(t) are in 24 time slots separately encoded by hour is the best.**
- RF model, according to CL2 configuration, has provided a
  - MAE(system) of 0.16, against the value of 0.4 of TFR PDE.
- The time needed for the LOOCV computation by data-driven models is lower if compared to TFR PDE.
- Different errors and improvements have been detected on specific sensors so that an additional phase of feature engineering for coding the context and road features could improve the solution



Structure of the course up to now …..

- Overview
- What is Symbolic, Classification
- Hybrid solutions
- Physically Informed, PINN
- Deep Reinforced Learning and Symbolic at Scale
- Knowledge ←→ NN
- From RAG LLM to Agentic LLM
- MLOps at Scale

SmartDS: Rules/decision-tree orchestrator + ML/NN tools

neuro-symbolic / hybrid, but **symbolic-dominant** because the *system-level decision policy* is explicitly encoded

Hybrid expert system (symbolic control, statistical inference at leaves)

- The hand-made decision tree (or rules/flowchart) is symbolic: discrete branching logic, explicitly designed, traceable.
- The leaf computations (calling tools, including NN predictions) are sub-symbolic / statistical components used as oracles.