# General Framework for Classification at the Top

**Ing. Václav Mácha**

September 20, 2023

**Supervisor:** doc. Ing. Václav Šmídl, Ph.D.
**Supervisor specialist:** Mgr. Lukáš Adam, Ph.D.

# Motivation

# Binary Classification

- General form of binary classification

$$\begin{aligned} \underset{\boldsymbol{w},\, t}{\text{minimize}} \quad & C_1 \sum_{i \in \mathcal{I}_-} \mathbb{1}_{[s_i \geq t]} + C_2 \sum_{i \in \mathcal{I}_+} \mathbb{1}_{[s_i < t]} \\ \text{subject to} \quad & s_i = f(\boldsymbol{x}_i; \boldsymbol{w}), \quad i \in \mathcal{I} \end{aligned}$$

- $\boldsymbol{x}_i \in \mathbb{R}^d$ is a sample and $y_i \in \{0, 1\}$ its corresponding label, $C_1, C_2 \in \mathbb{R}$ are constants
- $\mathcal{I} = \mathcal{I}_- \cup \mathcal{I}_+$ is a set of indices of all samples where

$$\begin{aligned} \mathcal{I}_- &= \{ i \mid i \in \{1, 2, \ldots, n\} \;\wedge\; y_i = 0 \} \\ \mathcal{I}_+ &= \{ i \mid i \in \{1, 2, \ldots, n\} \;\wedge\; y_i = 1 \} \end{aligned}$$

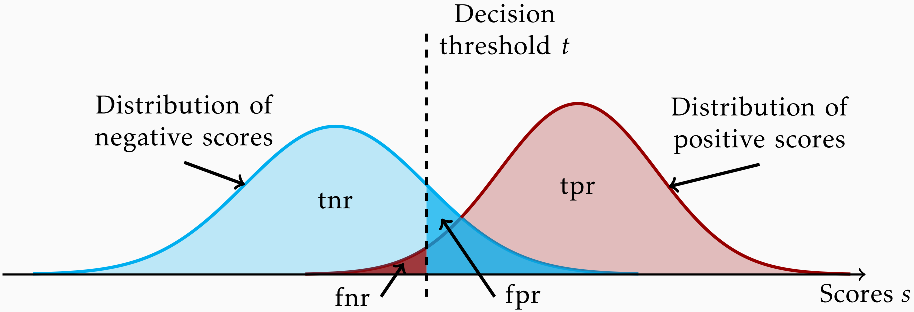
- $\mathbb{1}_{[\cdot]}$ is Iverson function defined by

$$\mathbb{1}_{[x]} = \begin{cases} 0 & \text{if } x \text{ is false} \\ 1 & \text{if } x \text{ is true} \end{cases}$$

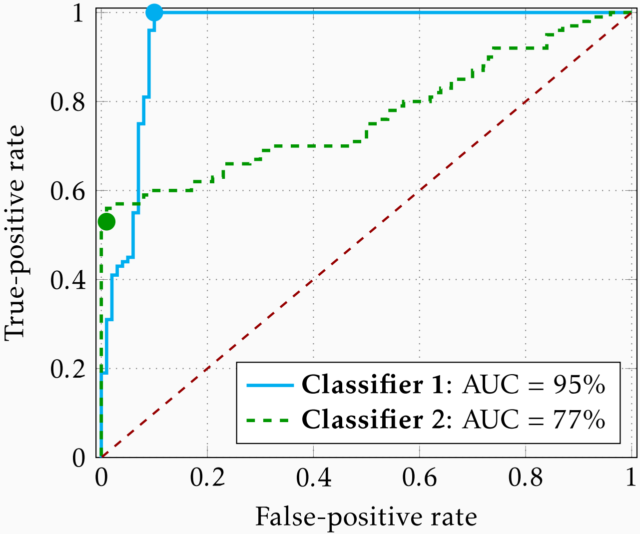
- Classifier consists of two parts: model $f : \mathbb{R}^d \mapsto \mathbb{R}$ with trainable parameters $\boldsymbol{w}$ that maps samples $\boldsymbol{x}$ to scores $s$ and a decision threshold $t \in \mathbb{R}$

# False rates

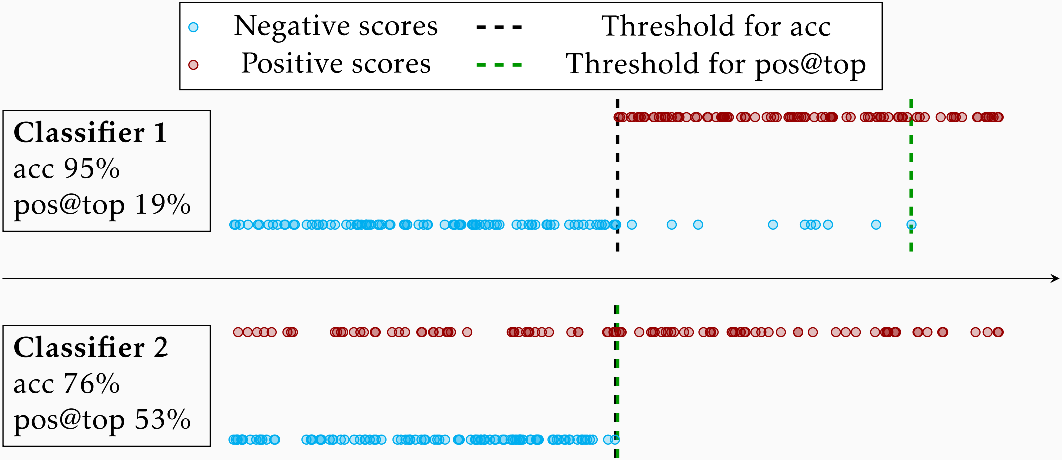
- Inference: Sample $\boldsymbol{x}$ is classified as positive if $s = f(\boldsymbol{x}; \boldsymbol{w}) \geq t$



# Classifier 1 is better ... or not?



# Sometimes Classifier 2 is the better one...



# Classification at the Top

# General problem formulation

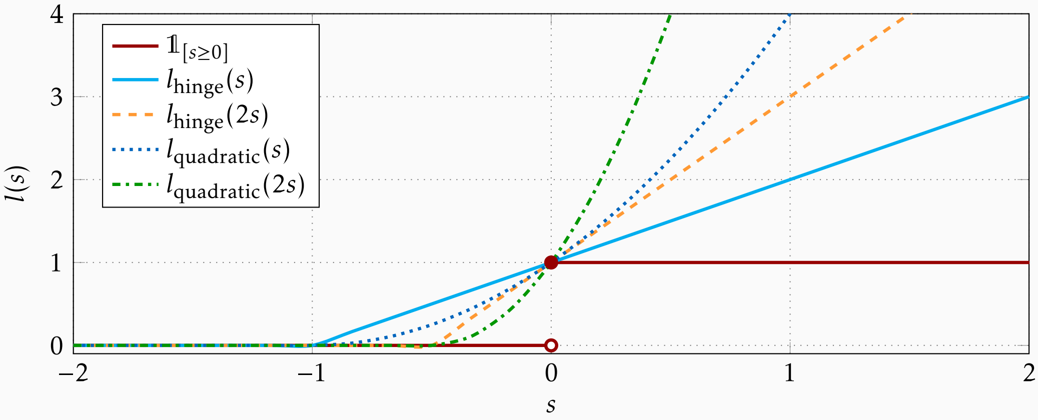
- **Goal:** correctly classify only the most relevant samples. The most relevant samples are samples with the highest scores
- General formulation

$$\begin{aligned} \operatorname*{minimize}_{\boldsymbol{w}} \quad & C_1 \sum_{i \in \mathcal{I}_-} \mathbb{1}_{[s_i \geq t]} + C_2 \sum_{i \in \mathcal{I}_+} \mathbb{1}_{[s_i < t]} \\ \text{subject to} \quad & s_i = f(\boldsymbol{x}_i; \boldsymbol{w}), \quad i \in \mathcal{I}, \\ & t = G(\boldsymbol{s}, \boldsymbol{y}) \end{aligned}$$

  where threshold $t$ is a function of all scores
- Difficult problem: constrained, discontinuous, generally non-convex, and non-decomposable

# How to get continuous objective function?

- By replacing $\mathbb{1}_{[\cdot]}$ Iverson function with its surrogate approximation



# General surrogate formulation

- Using the surrogate approximation to replace $\mathbb{1}_{[\cdot]}$ leads to the general surrogate formulation

$$
\begin{aligned}
\underset{\boldsymbol{w}}{\text{minimize}} \quad & C_1 \sum_{i \in \mathcal{I}_-} l(s_i - t) + C_2 \sum_{i \in \mathcal{I}_+} l(t - s_i) \\
\text{subject to} \quad & s_i = f(\boldsymbol{x}_i; \boldsymbol{w}), \quad i \in \mathcal{I}, \\
& t = G(\boldsymbol{s}, \boldsymbol{y})
\end{aligned}
$$

# How to choose the decision threshold?

- For simplicity, we focus only on formulations that minimize the false-negative rate and the threshold function $G$ depends only on negative samples

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \frac{1}{2}\left\|\boldsymbol{w}\right\|^2 + \frac{1}{n_+}\sum_{i\in\mathcal{I}_+} l(t - s_i) \\ \text{subject to} \quad & s_i = f(\boldsymbol{x}_i; \boldsymbol{w}), \quad i \in \mathcal{I}, \\ & t = G(\boldsymbol{s}, \boldsymbol{y}) \end{aligned}$$

where we use $C_1 = 0$ and $C_2 = \frac{1}{n_+}$. Regularization is added for numerical stability.

# How to choose the decision threshold?

- For simplicity, we focus only on formulations that minimize the false-negative rate and the threshold function $G$ depends only on negative samples

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n_+}\sum_{i\in\mathcal{I}_+} l(t - s_i) \\ \text{subject to} \quad & s_i = f(\boldsymbol{x}_i; \boldsymbol{w}), \quad i \in \mathcal{I}, \\ & t = G(\boldsymbol{s}, \boldsymbol{y}) \end{aligned}$$

  where we use $C_1 = 0$ and $C_2 = \frac{1}{n_+}$. Regularization is added for numerical stability.
- *TopPush* maximizes the number of positive samples at the top

$$t = G_{TopPush}(\boldsymbol{s}, \boldsymbol{y}) = \max_{j\in\mathcal{I}_-} s_j$$

## How to choose the decision threshold?

- For simplicity, we focus only on formulations that minimize the false-negative rate and the threshold function $G$ depends only on negative samples

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n_+}\sum_{i\in\mathcal{I}_+} l(t - s_i) \\ \text{subject to} \quad & s_i = f(\boldsymbol{x}_i;\boldsymbol{w}), \quad i \in \mathcal{I}, \\ & t = G(\boldsymbol{s}, \boldsymbol{y}) \end{aligned}$$

  where we use $C_1 = 0$ and $C_2 = \frac{1}{n_+}$. Regularization is added for numerical stability.
- *TopPush* maximizes the number of positive samples at the top

$$t = G_{TopPush}(\boldsymbol{s}, \boldsymbol{y}) = \max_{j\in\mathcal{I}_-} s_j$$

- *Pat&Mat-NP* maximizes true-positive rate with fixed false-positive rate

$$t = G_{Pat\&Mat-NP}(\boldsymbol{s}, \boldsymbol{y}) \quad \Longleftrightarrow \quad t \ \text{ solves } \ \frac{1}{n_-}\sum_{i\in\mathcal{I}_-} l(s_i - t) = \tau$$

# Classification at the Top: Linear Model

# Linear Model

- General surrogate formulation with linear model $f(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x}$

$$
\begin{aligned}
\underset{\boldsymbol{w}}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n_+}\sum_{i \in \mathcal{I}_+} l(t - s_i) \\
\text{subject to} \quad & s_i = \boldsymbol{w}^\top \boldsymbol{x}_i, \quad i \in \mathcal{I}, \\
& t = G(\boldsymbol{s}, \boldsymbol{y})
\end{aligned}
$$

- Properties that we are interested in:
    - Convexity of the objective function
    - Robustness to outliers

# Convexity of the objective function

**Theorem**

If the threshold $t$ is a convex function of the weights $\boldsymbol{w}$, then function

$$L(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n_+}\sum_{i \in \mathcal{I}_+} l(t(\boldsymbol{w}) - \boldsymbol{w}^\top \boldsymbol{x}_i)$$

is convex.

- What does it mean?
  - Both formulations *TopPush* and *Pat&Mat-NP* have convex thresholds
  - Both formulations are convex and continuous
  - We can solve both formulations using gradient descent algorithm

## How to solve it?

- Using gradient descent

$$\boldsymbol{w}^{k+1} \leftarrow \boldsymbol{w}^{k} - \alpha^{k} \cdot \nabla L(\boldsymbol{w}^{k}),$$

where $\alpha^k > 0$ is a learning rate, and $\nabla L(\boldsymbol{w}^k)$ is a gradient of the objective function

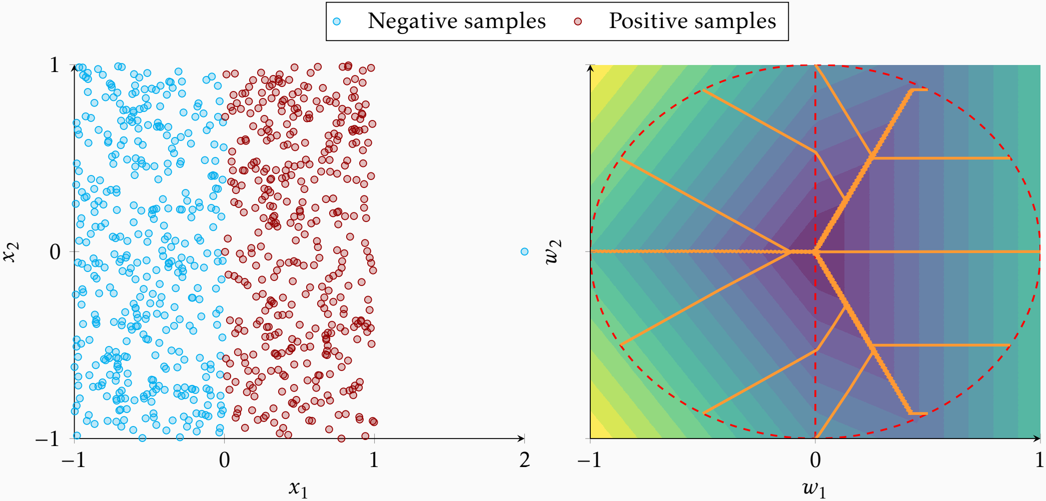
$$\nabla L(\boldsymbol{w}) = \boldsymbol{w} + \frac{1}{n_+} \sum_{i \in \mathcal{I}_+} l' \left( t(\boldsymbol{w}) - \boldsymbol{w}^\top \boldsymbol{x}_i \right) \left( \nabla t(\boldsymbol{w}) - \boldsymbol{x}_i \right)$$

- How to compute gradient of the threshold $\nabla t(\boldsymbol{w})$?
  - For *TopPush* it is easy

$$j^\star = \operatorname*{arg\,max}_{j \in \mathcal{I}_-} s_j \quad \rightarrow \quad t = s_{j^\star} \quad \rightarrow \quad \nabla t(\boldsymbol{w}) = \nabla f(\boldsymbol{x}_{j^\star}; \boldsymbol{w}) = \boldsymbol{x}_{j^\star}$$

  - For *Pat&Mat-NP* we have to use **implicit function theorem**.

# When convexity is not enough...



# Classification at the Top: Non-linear Model

# Non-linear Model

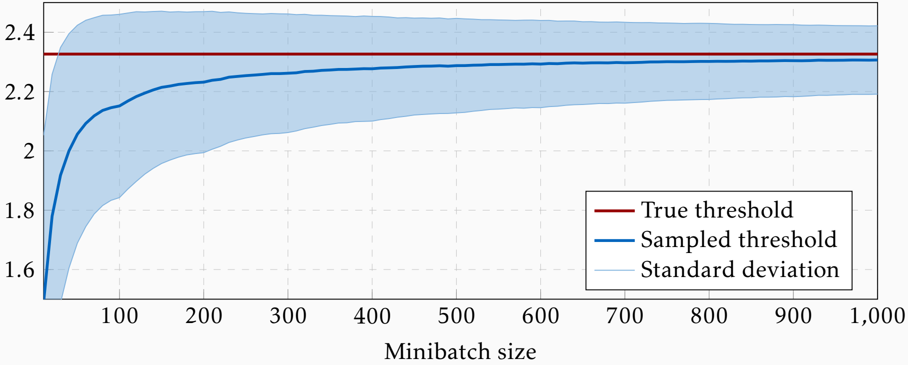
- General surrogate formulation with non-linear model $f(\boldsymbol{x}; \boldsymbol{w})$

$$
\begin{aligned}
\underset{\boldsymbol{w}}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n_+}\sum_{i \in \mathcal{I}_+} l(t - s_i) \\
\text{subject to} \quad & s_i = f(\boldsymbol{x}_i; \boldsymbol{w}), \quad i \in \mathcal{I}, \\
& t = G(\boldsymbol{s}, \boldsymbol{y})
\end{aligned}
$$

- Disadvantages:
  - Objective function is not convex
  - Non-linear models are usually large and expensive to train
- What to do if the dataset is too large to fit in memory? Stochastic gradient descent.

# Issues with stochastic gradient descent

- The threshold is a function of all scores $\rightarrow$ the loss function is non-decomposable
- As a result, stochastic gradient descent provides a biased gradient estimate

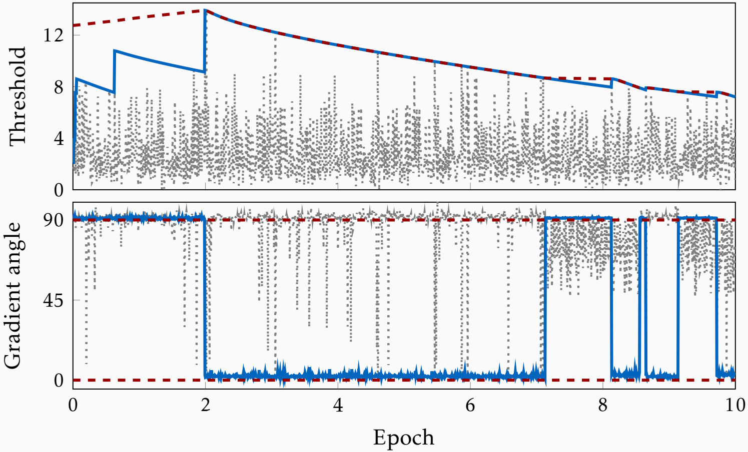



- How to reduce bias? Increase size of minibatch ...

# Is there a better way to reduce bias?

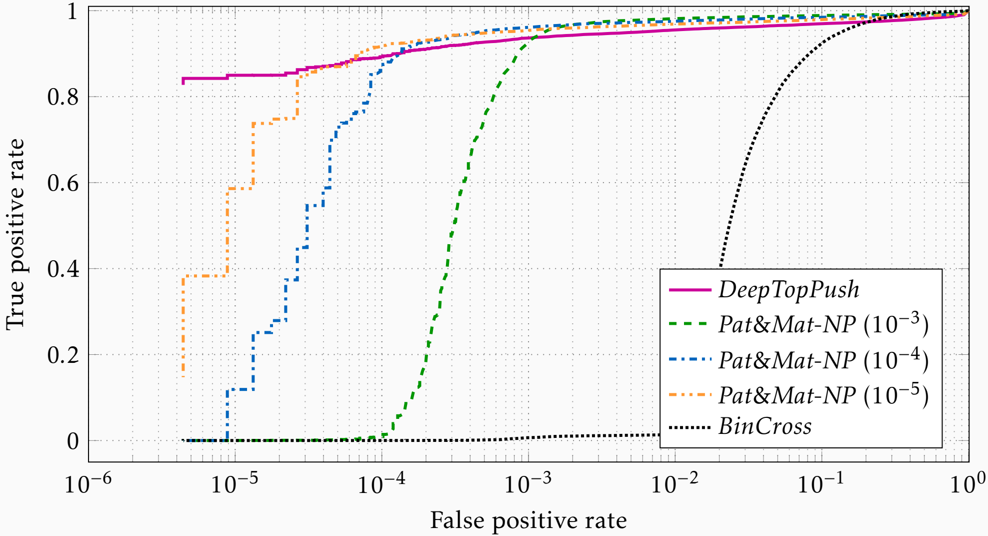
- *DeepTopPush*: Add threshold from last minibatch

$$j^\star = \arg\max_{j \in \mathcal{I}_-} s_j \quad \rightarrow \quad t = s_{j^\star} \quad \rightarrow \quad \nabla t(\boldsymbol{w}) = \nabla f(\boldsymbol{x}_{j^\star}; \boldsymbol{w})$$

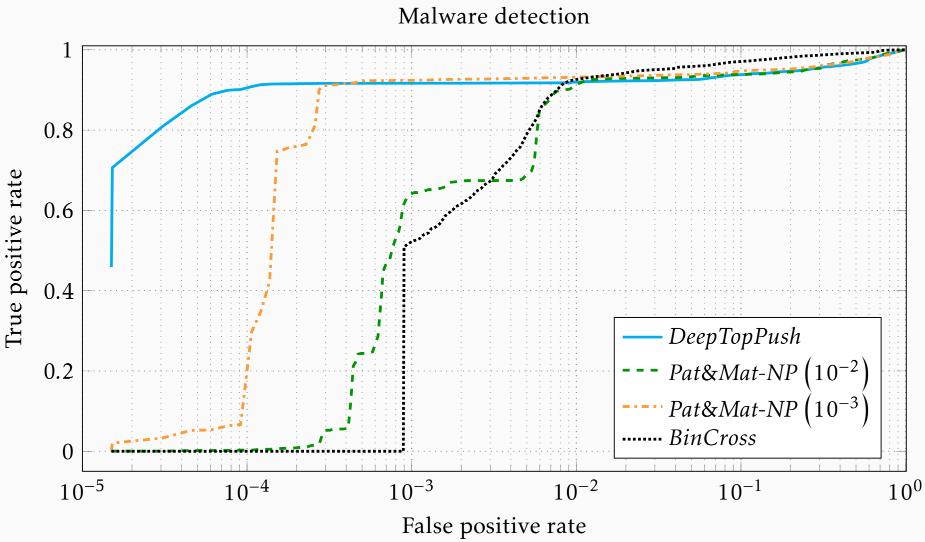


# How does it work?

# Steganalysis



# AVAST: Malware detection



# Main contributions

# Contributions

- **Unification Contributions:**
  - Introduction of a unified framework for Classification at the Top
  - Showed that problems such as Ranking or Accuracy at the Top fall into the framework
  - Introduction of *Pat&Mat* and *Pat&Mat-NP* formulations
- **Theoretical Contributions:**
  - Derivation of theoretical properties of formulations from the framework with linear model
  - Derivation of dual forms and use of non-linear kernels
- **Algorithmic Contributions:**
  - Derivation of an efficient algorithm for solving dual forms
  - Introduction of a modified stochastic gradient descent
  - Introduction of *DeepTopPush* formulation

# Thank you for your attention.