

Efficient Loss-Based Decoding on Graphs for Extreme Classification

Itay Evron, Edward Moroshko, and Koby Crammer

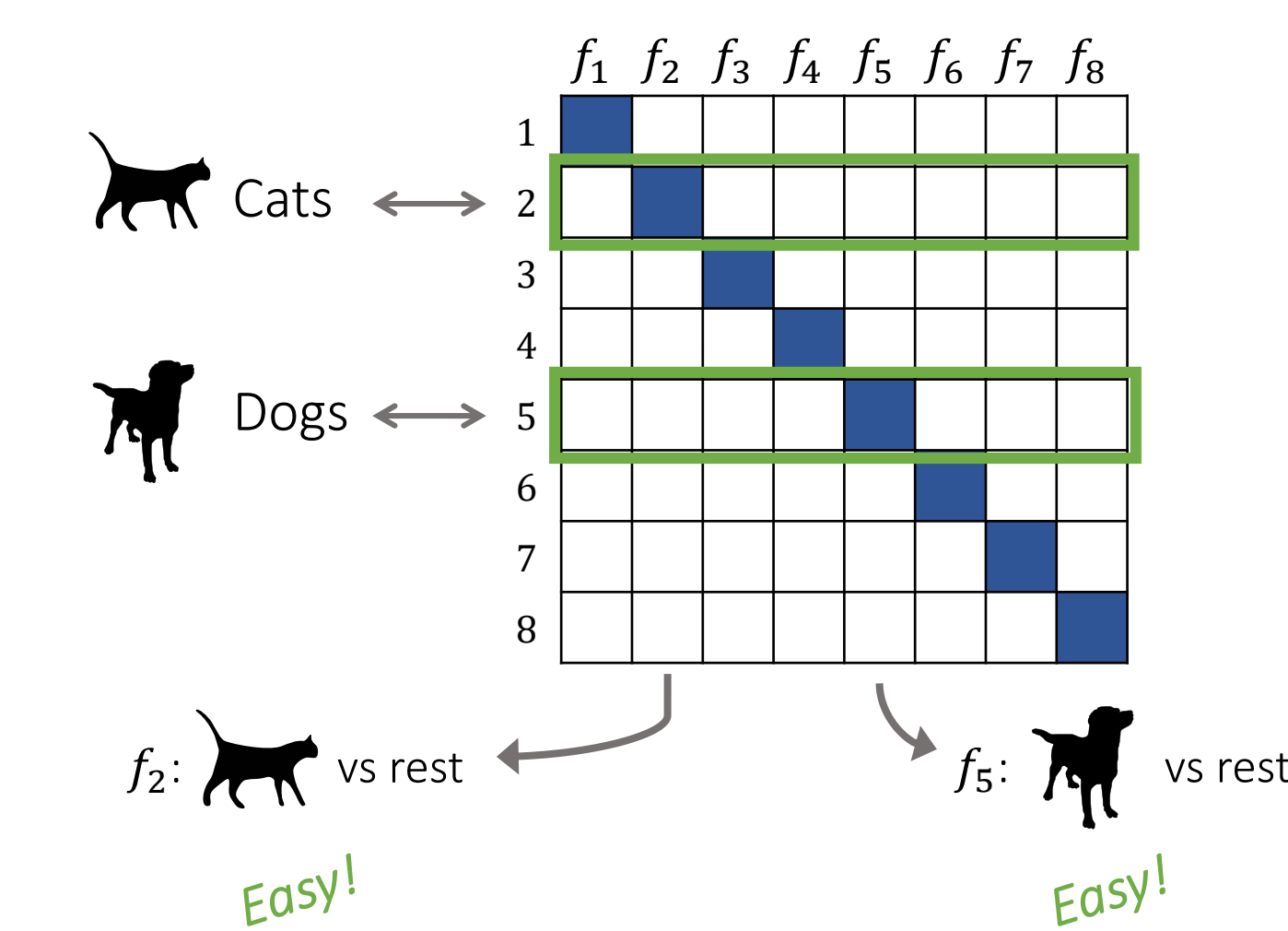
Extreme multiclass classification

- Tasks with an extremely large number of classes K .
 - Time and space complexities during training and inference become critical.
 - Datasets are typically sparse, i.e. samples have on average only $d_{nnz} \ll d$ nonzero features.
- We propose a graph-based classification scheme with time and space complexities logarithmic in K .

One vs Rest – Simple but expensive

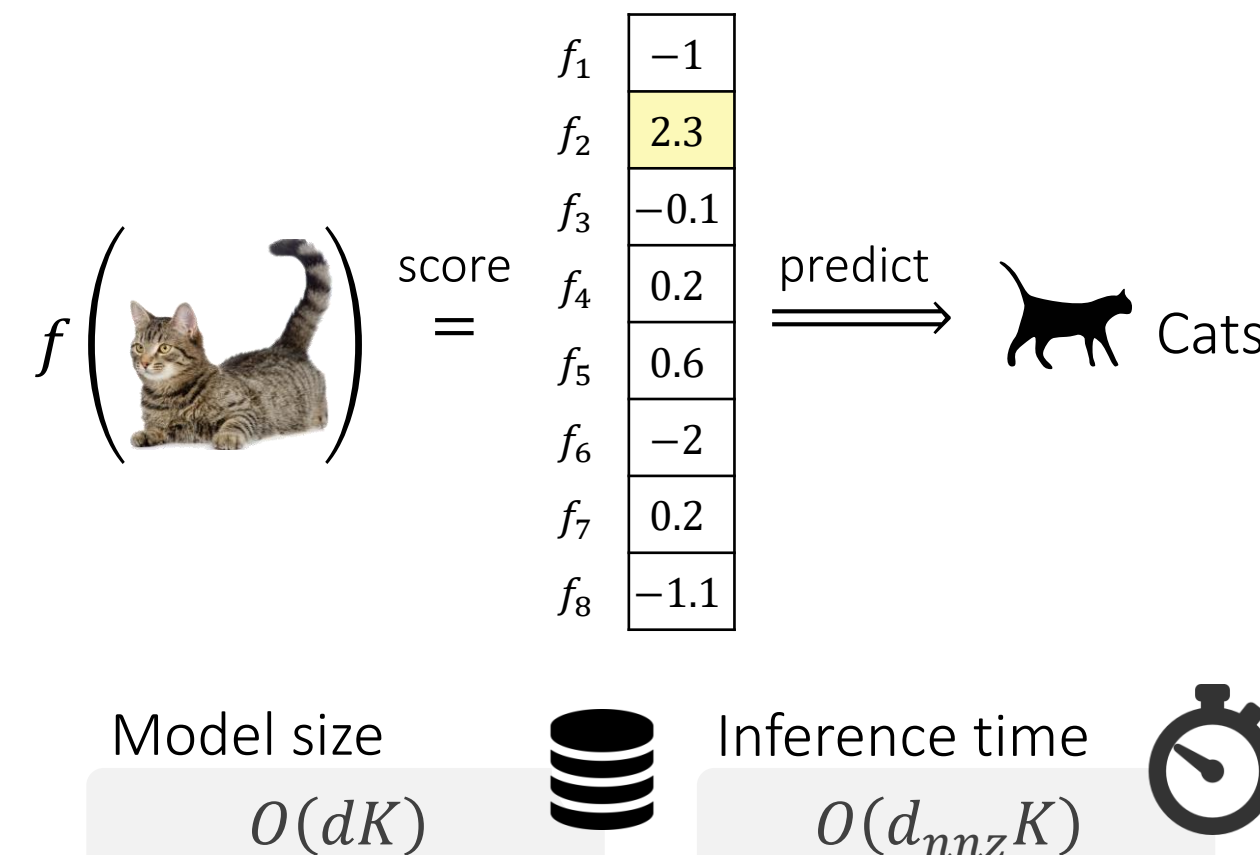
Training:

- Learn K binary classifiers $f_1, \dots, f_K: \mathcal{X} \rightarrow \mathbb{R}$



Inference for $x \in \mathcal{X}$:

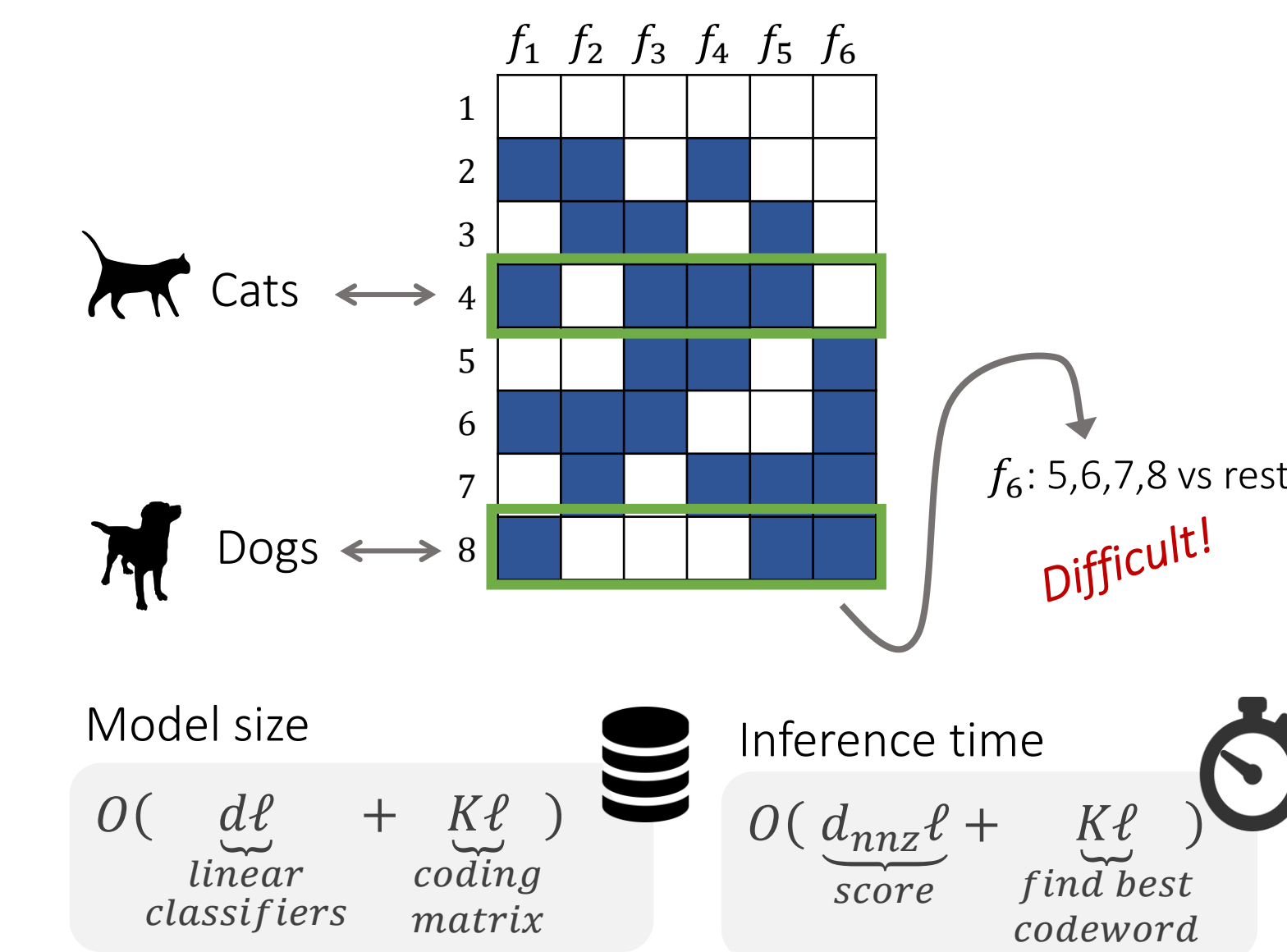
- Score all: $f(x) = [f_1(x), \dots, f_K(x)]$
- Predict: $\hat{y} = \arg \max_k f_k(x)$



Error Correcting Output Coding

Training:

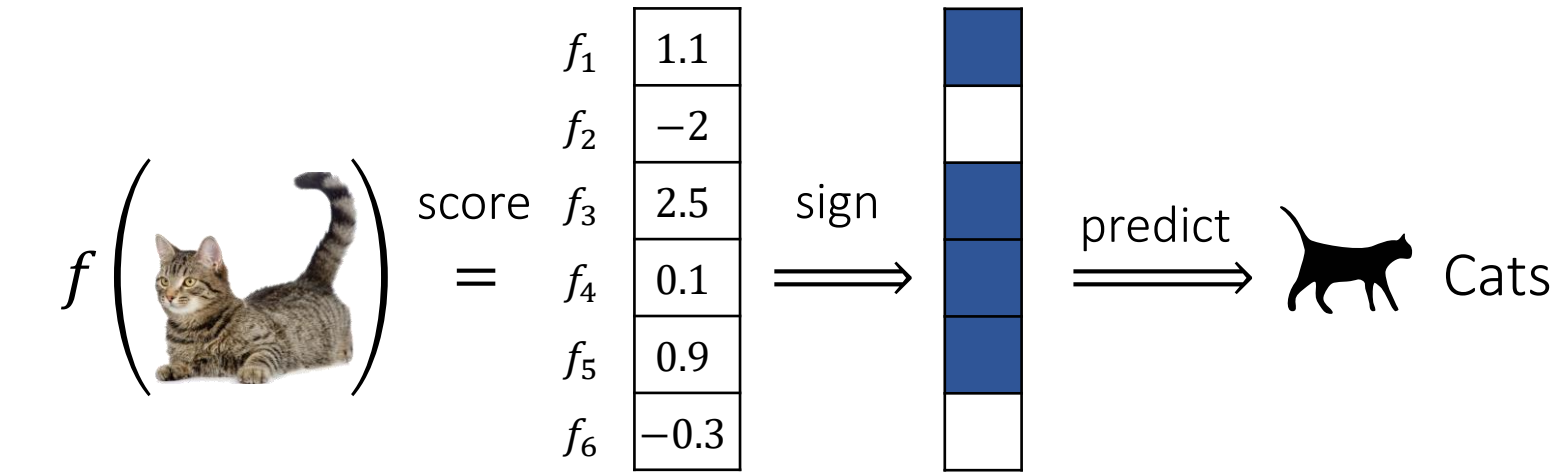
- Get a coding matrix $M \in \{-1, 1\}^{K \times \ell}$
- Map each codeword to a class (e.g. arbitrarily)
- Learn ℓ binary classifiers $f_1, \dots, f_\ell: \mathcal{X} \rightarrow \mathbb{R}$



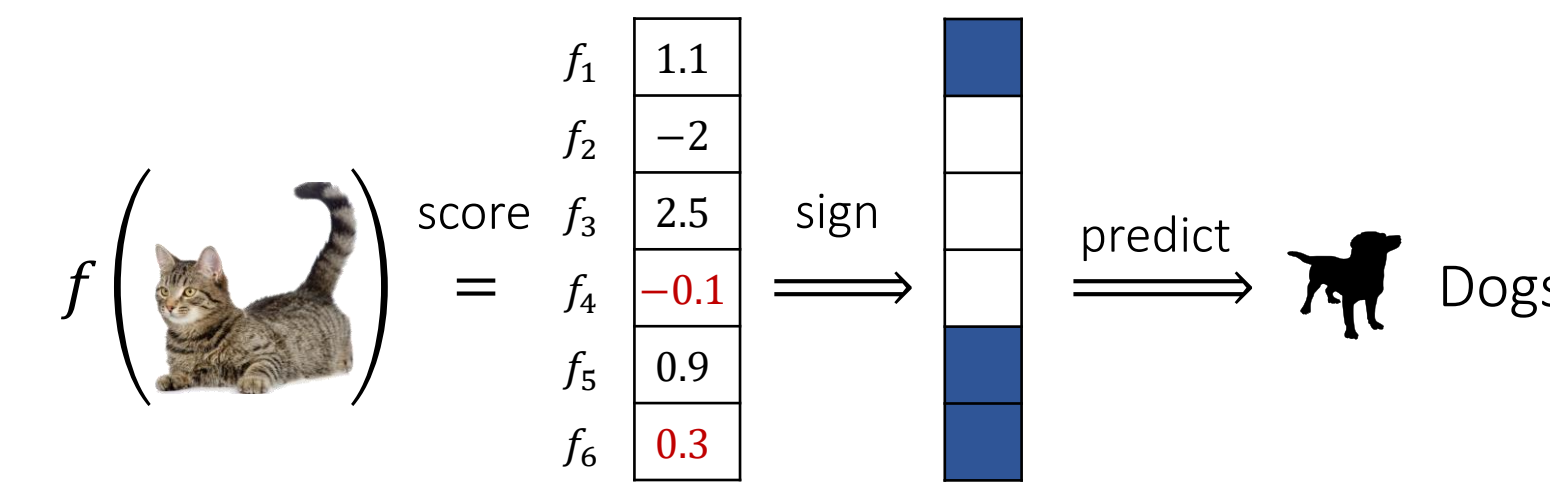
Inference for $x \in \mathcal{X}$:

- Score all: $f(x) = [f_1(x), \dots, f_\ell(x)]$
- Predict: $\hat{y} = \arg \min_k d_{\text{Ham}}(M_k, \text{sign}(f(x)))$

Inference example:



Mistake example:



Loss based decoding

- Instead of minimizing the Hamming distance,

$$\text{predict } \hat{y} = \arg \min_k \sum_{j=1}^{\ell} \mathcal{L}(M_{k,j} \times f_j(x))$$

- An upper bound of the training multiclass error is proportional to:

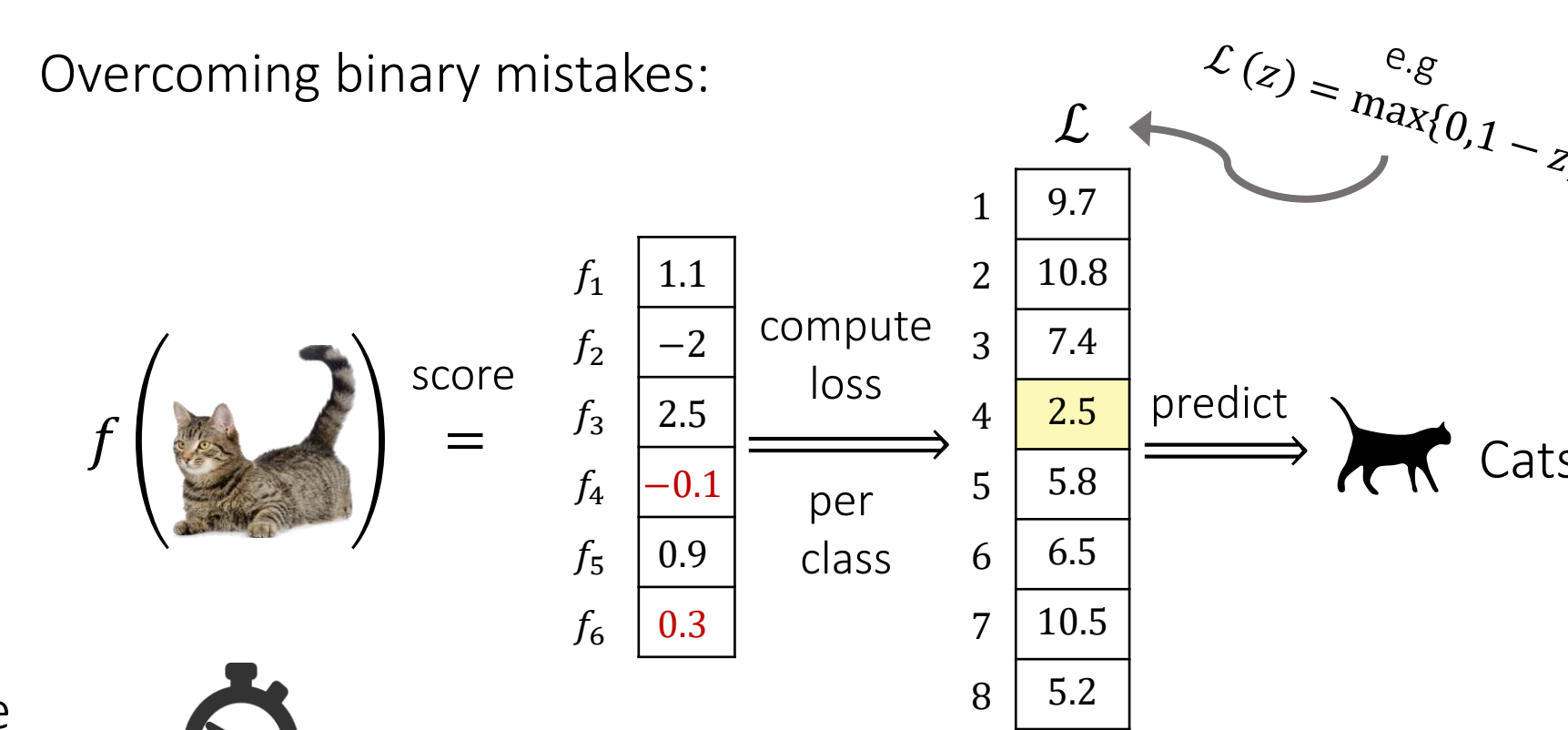
$$\text{Number of predictors} \sim \frac{\ell \times \varepsilon}{\rho} \quad \text{Average binary loss}$$

$$\text{Minimum row distance} \sim \frac{\rho}{\rho}$$

- The decoding loss function matters.

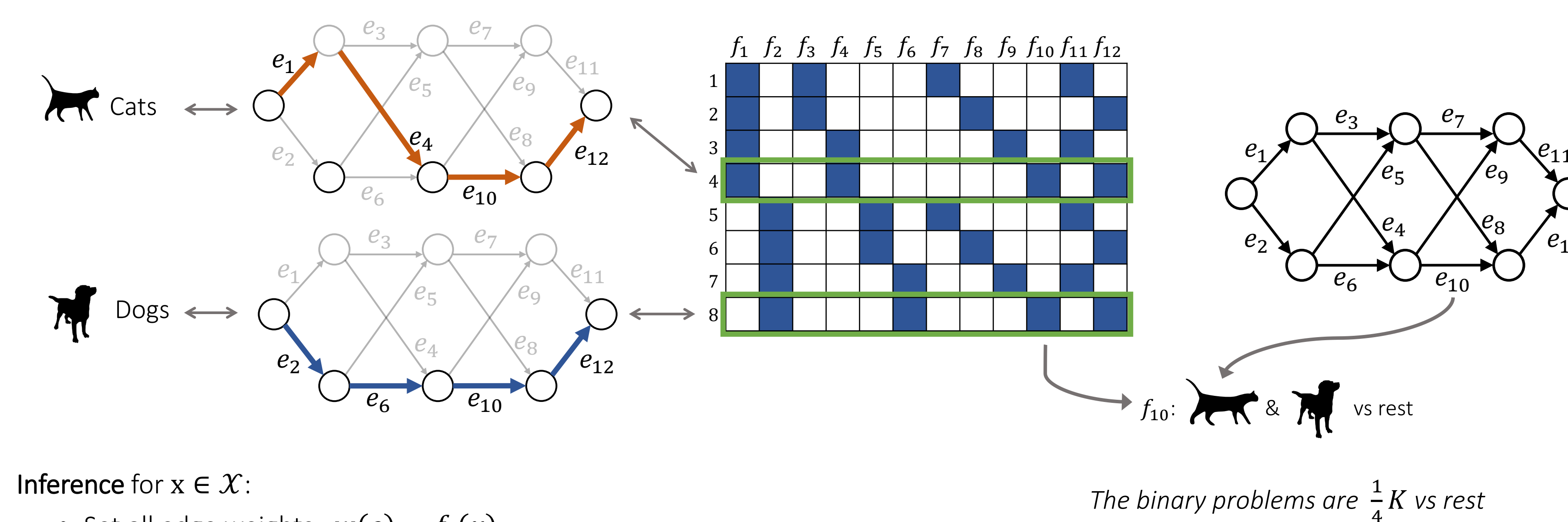
$$\text{Inference time} \sim O\left(\frac{d_{nnz}\ell}{\text{score}} + \frac{K\ell}{\text{loss based decoding}}\right)$$

Overcoming binary mistakes:



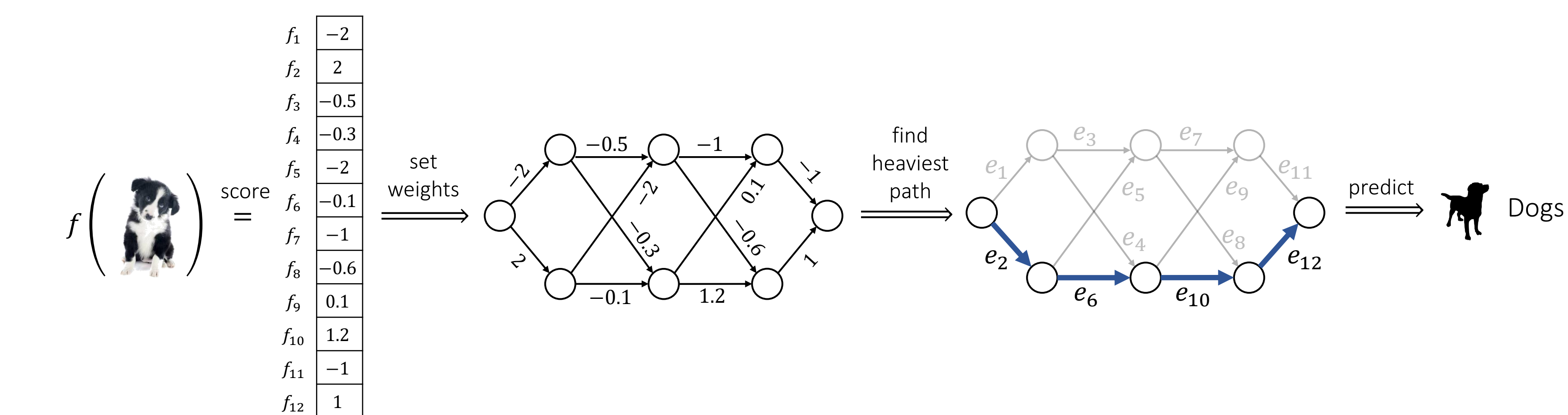
Our model – Wide-LTLS

- Based on LTLS [Jasinska and Karampatziakis 2016].
- Build a trellis graph with exactly K paths.
- Map each path to a class (e.g. arbitrarily).
- For each edge $e \in E$:
 - Train a classifier $f_e: \mathcal{X} \rightarrow \mathbb{R}$ on the entire training set:
 - Separate classes (=paths) that use this edge, from the classes that do not.



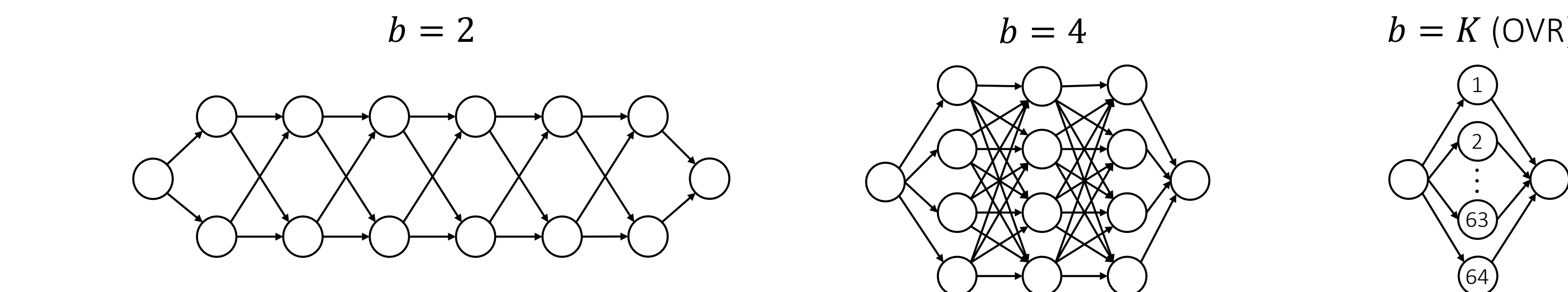
Inference for $x \in \mathcal{X}$:

- Set all edge weights: $w(e) = f_e(x)$.
- Find the heaviest path and predict its class.



Graph width controls complexity

- The following graphs have $K = 64$ paths (=classes), but different graph widths b



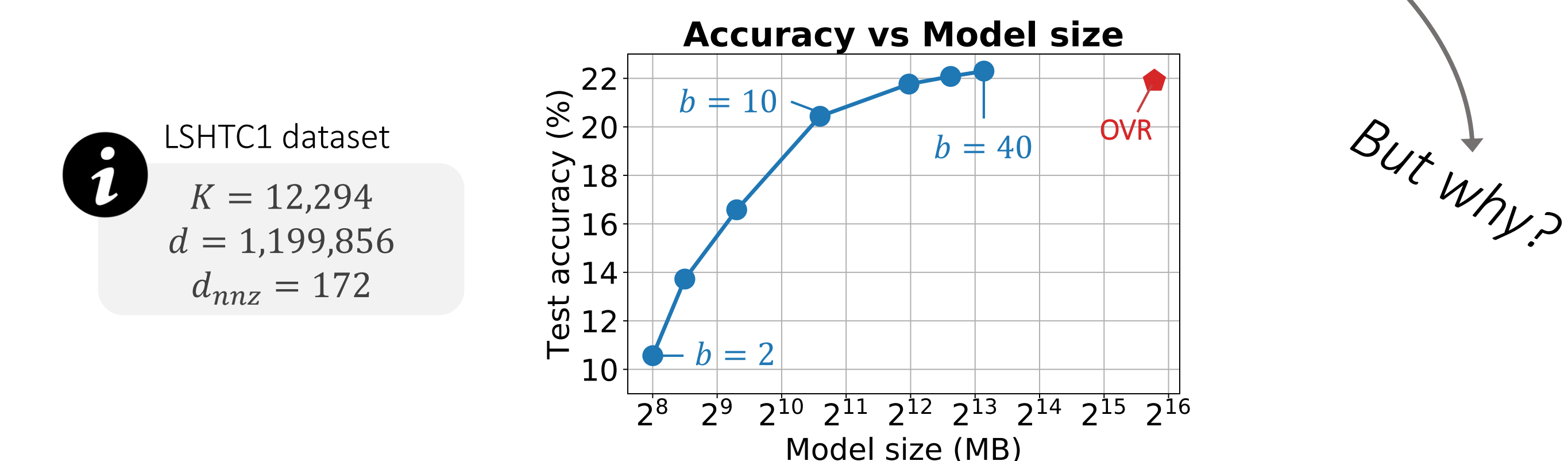
- The number of edges / classifiers is: $|E| = O\left(\frac{b^2}{\log b} \cdot \log K\right)$.
- Therefore,

$$\text{Model size} \sim O\left(d \cdot \frac{b^2}{\log b} \cdot \log K\right)$$

$$\text{Inference time (for } b < \sqrt{K}) \sim O\left(d_{nnz} \cdot \frac{b^2}{\log b} \cdot \log K\right)$$

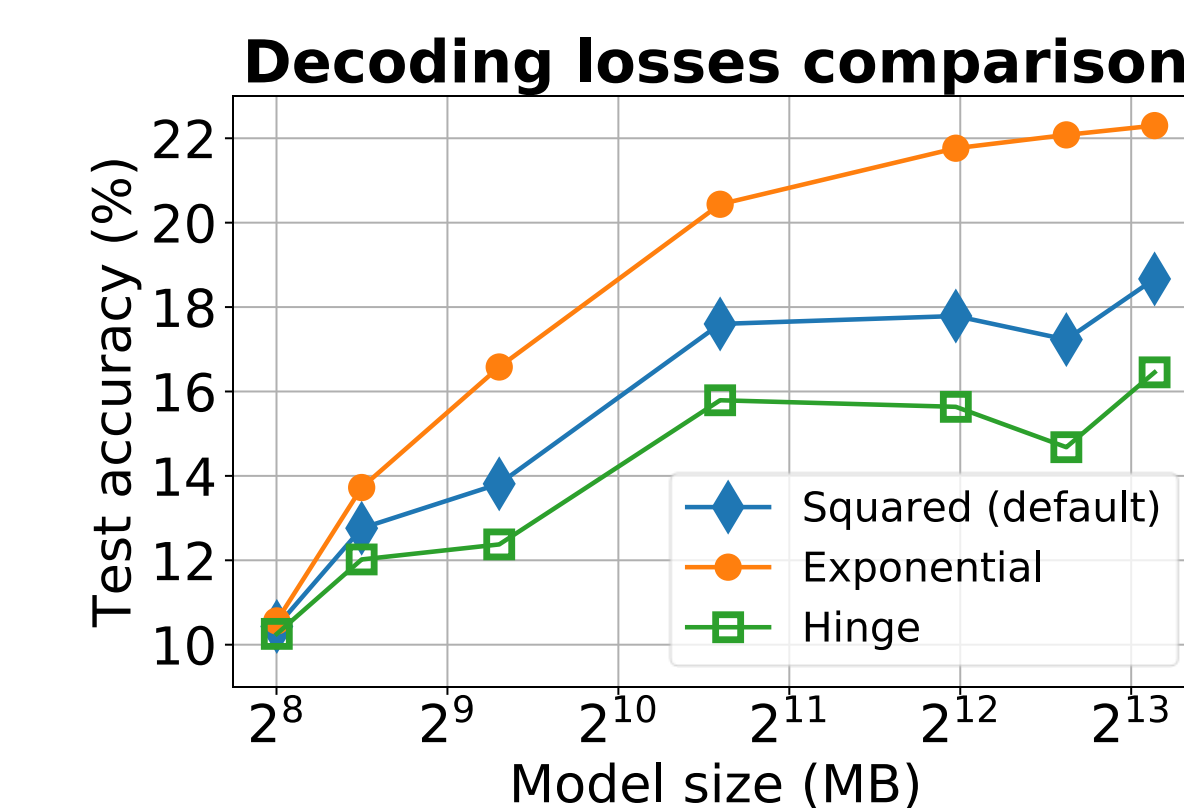
Graph width controls performance

- Our model offers a tradeoff between accuracy and model size.



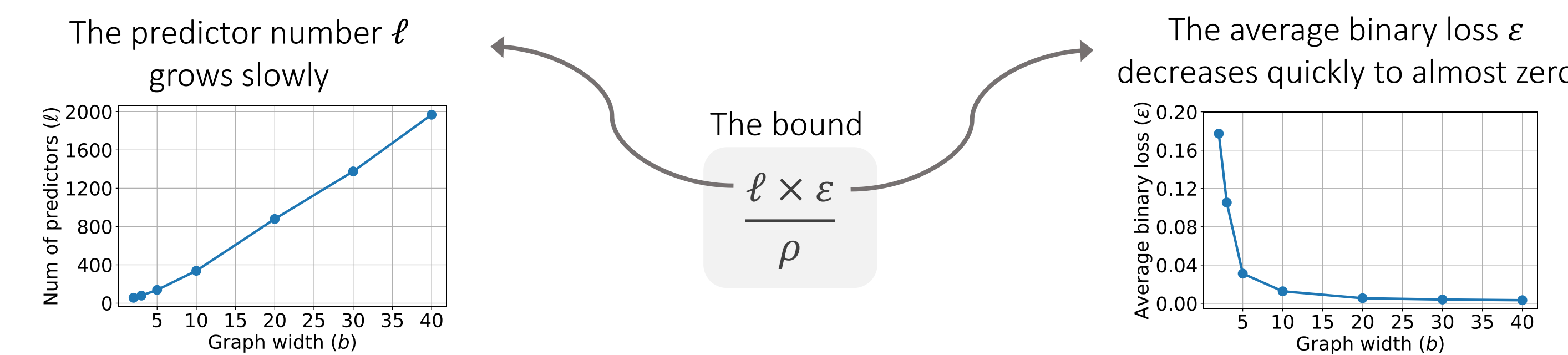
W-LTLS as loss-based decoding

- We prove that W-LTLS performs loss-based decoding with the squared loss $\mathcal{L}(z) = (1 - z)^2$.
- We show how to generalize W-LTLS to any loss function \mathcal{L} , and perform loss based decoding in time logarithmic in K .
- The decoding loss function matters!
 - The loss function can be chosen quickly after training.

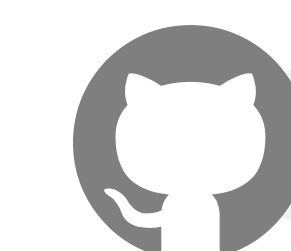
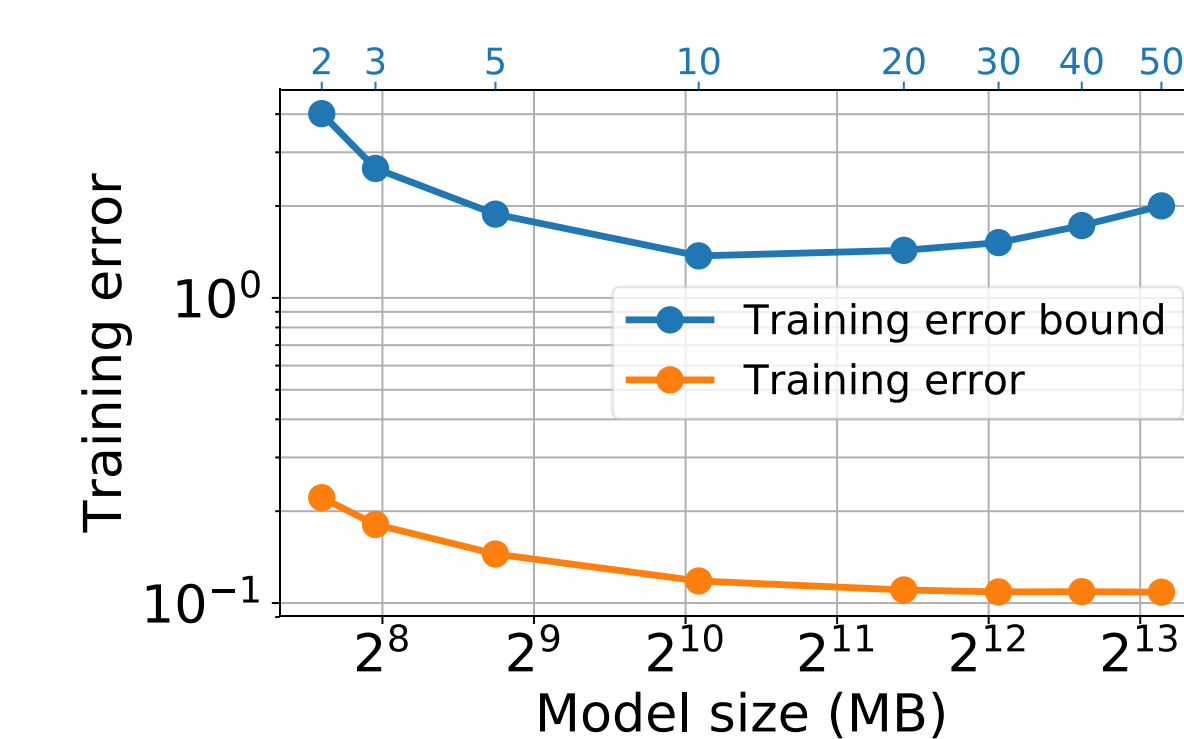


Wider graph – Easier binary problems

- The subproblems are $\frac{1}{b^2}K$ -vs-rest, thus get easier.



The bound can help find a sweet spot.



<https://github.com/ievron/wltls/>, arXiv: 1803.03319