Teaching old sums new tricks: Data-driven enhancements for elliptic curve rank heuristics

Representation Theory XIX
Dubrovnik, Croatia
June 26th, 2025

Zvonimir Bujanović[1], Matija Kazalicki[1], Domagoj Vlah[2]

[1]University of Zagreb, Faculty of Science, Department of Mathematics
[2]University of Zagreb, Faculty of Electrical Engineering and Computing, Department of Applied Mathematics

Outline

- Ranks of elliptic curves
- Rank classification via multi-value Mestre-Nagao sums
- Learning optimal Mestre-Nagao sums

Ranks of elliptic curves

Rational points on elliptic curve

Definition
For $a_1, a_2, a_3, a_4, a_6 \in \mathbb{Q}$

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

is called **elliptic curve** over $\mathbb{Q}$.
(assume the discriminant $\Delta(a_1, a_2, a_3, a_4, a_6)$ is nonzero)

Rational points on elliptic curve

Definition
For $a_1, a_2, a_3, a_4, a_6 \in \mathbb{Q}$

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

is called **elliptic curve** over $\mathbb{Q}$.
(assume the discriminant $\Delta(a_1, a_2, a_3, a_4, a_6)$ is nonzero)

N - the conductor of the elliptic curve E

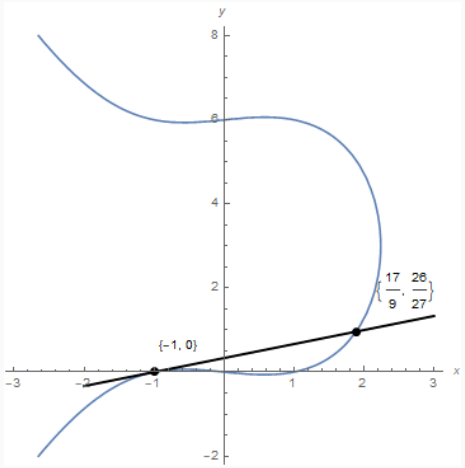
Rational points on elliptic curve

Definition
For $a_1, a_2, a_3, a_4, a_6 \in \mathbb{Q}$

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

is called **elliptic curve** over $\mathbb{Q}$.
(assume the discriminant $\Delta(a_1, a_2, a_3, a_4, a_6)$ is nonzero)

N - the conductor of the elliptic curve E
$E(\mathbb{Q})$ - the group of rational points on E



Rank of elliptic curve

Theorem (Mordell)
$E(\mathbb{Q})$ *is a* **finitely generated** *abelian group, i.e.*

$$E(\mathbb{Q}) \cong E_{tors}(\mathbb{Q}) \times \mathbb{Z}^r,$$

$E_{tors}(\mathbb{Q})$ *- a finite group of elements of finite order*

r *- a non-negative integer - (algebraic)* **rank** *of the elliptic curve* E

Elliptic curves of high rank

- Rank is a mysterious quantity. Is the rank unbounded? Current record is 29 (Elkies - Klagsbrun).

Rank 29 example

$$y^2 + xy = x^3 -$$
$$270061832416309222184346521452974537847680546218363579547373 85x +$$
$$552580585513423764757366995911181918215210670325350796083724047791494132777161734256367214 97$$

Rank 29 example

$$y^2 + xy = x^3 -$$

$$270061832416309222184346521452974537847680546218363579547373 85x +$$

$$552580585513423764757366995911181918215210670325350796083724047791494132777161734256367214 97$$

P1 = [289119547422853718945825553663 4, 1159930748096124706459835910727318679593425283]
P2 = [340254216532212781145148464223 4, 1661508223164691055862657623730465560755290883]
P3 = [429876002655846724042210756479 4, 4313142249890236204790986787384907722927474563]
P4 = [372875666777094700988445571455 4, 2530180219584734091116528693531660545660397443]
P5 = [599174413205207823051118513023 4, 1041890162884203436230127305572830066921885 8883]
P6 = [323649353463276852054022722303 4, 1324626796262167243658687198416201825373745283]
P7 = [782266861349911742323806893862 34,690394210062759896503429654125516779999512554883]
P8 = [114926056435488593746356051402 34, 3553631691145095215546162423830845602961894 0883]
P9 = [-514330336238422980490608811856 6, 7622356511107986864120352355674305680222368483]
P10 = [443985655575065435281568435002, 6584468124388858623214803939643557365635620355]
P11 = [-979565018904269680752629749766, 898734842210453768496670643871403863383217 0883]
P12 = [518489428521217824956646126183 4, 7390536788003150201273204464695859875054804 83]
P13 = [-446917102368714650206717961216 6, 9310658892841458934133221137392081403414455683]
P14 = [360640583511092548245052297023 4, 2183644666981703632482662193390480040127898883]
P15 = [161517445767853177326889931622 34,619088820924723389465199092764558314637472108 83]
P16 = [357368435594376638796236286975 4, 2094467155115749424853047283659077805560259203]
P17 = [-759376049938858166436491644166, 867917113545819719591402416180006181095211 9683]
P18 = [-532805871993588618210600311936 6, 6920588147379497633202935557367499676224350083]
P19 = [538026847489537735558303969455 4, 8105660240030025092450118297303424395856037443]
⋮

P29 = [242877826327752195954304393023 4, 1998325023610603606161737305486867803334410883]

Computing rank is hard

- Determining the rank of an elliptic curve is a computationally expensive task
- Finding rational points on elliptic curves is a difficult problem

Computing rank is hard

- Determining the rank of an elliptic curve is a computationally expensive task
- Finding rational points on elliptic curves is a difficult problem
- A descent algorithm commonly used eventually reduces to a naive point search on some auxiliary curves.
- There is no algorithm known to efficiently compute the rank for all curves.

Computing rank is hard

- Determining the rank of an elliptic curve is a computationally expensive task
- Finding rational points on elliptic curves is a difficult problem
- A descent algorithm commonly used eventually reduces to a naive point search on some auxiliary curves.
- There is no algorithm known to efficiently compute the rank for all curves.

Is it possible to determine rank without finding explicit generators?

Birch and Swinnerton-Dyer conjecture

Define $a_p = p + 1 - \#E(\mathbb{F}_p)$, for every prime p.

Birch and Swinnerton-Dyer conjecture

Define $a_p = p + 1 - \#E(\mathbb{F}_p)$, for every prime p.

Define **Hasse-Weil** L-function by absolutely convergent infinite product

$$L(E, s) = \prod_{p \text{ prime}} \left(1 - a_p p^{-s} + p^{1-2s}\right)^{-1}, \quad \text{for } \Re(s) > 3/2.$$

By the Modularity theorem $L(E, s)$ extends to an entire function.

Birch and Swinnerton-Dyer conjecture

Define $a_p = p + 1 - \#E(\mathbb{F}_p)$, for every prime p.

Define **Hasse-Weil** L-function by absolutely convergent infinite product

$$L(E, s) = \prod_{p \text{ prime}} \left(1 - a_p p^{-s} + p^{1-2s}\right)^{-1}, \quad \text{for } \Re(s) > 3/2.$$

By the Modularity theorem $L(E, s)$ extends to an entire function.

Conjecture (BSD)
Order of vanishing of $L(E, s)$ at $s = 1$ (the quantity known as analytic rank) is equal to the rank of elliptic curve E.

Heuristics for computing rank

An alternative approach to descent algorithms:

- Use rank heuristics that are inspired by BSD conjecture.
- These heuristics (we will call them Mestre-Nagao sums) help in identifying probable candidates for elliptic curves of high rank.

Heuristics for computing rank

An alternative approach to descent algorithms:

- Use rank heuristics that are inspired by BSD conjecture.
- These heuristics (we will call them Mestre-Nagao sums) help in identifying probable candidates for elliptic curves of high rank.
- In this work we consider two different heuristics:

$$S_0(B) = \frac{1}{\log B} \sum_{\substack{p<B, \\ \text{good reduction}}} \frac{a_p(E)\log p}{p},$$

Heuristics for computing rank

An alternative approach to descent algorithms:

- Use rank heuristics that are inspired by BSD conjecture.
- These heuristics (we will call them Mestre-Nagao sums) help in identifying probable candidates for elliptic curves of high rank.
- In this work we consider two different heuristics:

$$S_0(B) = \frac{1}{\log B} \sum_{\substack{p<B, \\ \text{good reduction}}} \frac{a_p(E)\log p}{p},$$

$$S_5(B) = \sum_{\substack{p<B, \\ \text{good reduction}}} \log\left(\frac{p+1-a_p(E)}{p}\right).$$

Heuristics for computing rank

An alternative approach to descent algorithms:

- Use rank heuristics that are inspired by BSD conjecture.
- These heuristics (we will call them Mestre-Nagao sums) help in identifying probable candidates for elliptic curves of high rank.
- In this work we consider two different heuristics:

$$S_0(B) = \frac{1}{\log B} \sum_{\substack{p<B, \\ \text{good reduction}}} \frac{a_p(E)\log p}{p},$$

$$S_5(B) = \sum_{\substack{p<B, \\ \text{good reduction}}} \log\left(\frac{p+1-a_p(E)}{p}\right).$$

- E has good reduction at $p \iff p \nmid \Delta_E$

Asymptotics of S_0 and S_5

- $\lim_{B\to\infty} S_0(B) = -r_{an} + 1/2$ if the limit exists, where r_{an} is the analytic rank of $E/\mathbb{Q}$ (Kim and Murty).

Asymptotics of S_0 and S_5

- $\lim_{B\to\infty} S_0(B) = -r_{an} + 1/2$ if the limit exists, where r_{an} is the analytic rank of $E/\mathbb{Q}$ (Kim and Murty).
- $S_5(B) = \log A + r \log\log B + o(1)$, where A is a constant that does not depend on B (Goldfeld).

Using Mestre-Nagao Sums to Find High-Rank Curves

How to identify curves with high rank?

1. Examine a family of elliptic curves already known to have moderately high ranks.
2. Compute $S_5(B)$ for a large sample of curves within the family.
3. Filter the curves by computed $S_5(B)$ using some threshold.
4. Apply descent algorithm for computing rank on filtered curves.

Using Mestre-Nagao Sums to Find High-Rank Curves

How to identify curves with high rank?

1. Examine a family of elliptic curves already known to have moderately high ranks.
2. Compute $S_5(B)$ for a large sample of curves within the family.
3. Filter the curves by computed $S_5(B)$ using some threshold.
4. Apply descent algorithm for computing rank on filtered curves.

Descent algorithms are computationally demanding. Computing $S_5(B)$ is much less so.

Using Mestre-Nagao Sums to Find High-Rank Curves

How to identify curves with high rank?

1. Examine a family of elliptic curves already known to have moderately high ranks.
2. Compute $S_5(B)$ for a large sample of curves within the family.
3. Filter the curves by computed $S_5(B)$ using some threshold.
4. Apply descent algorithm for computing rank on filtered curves.

Descent algorithms are computationally demanding. Computing $S_5(B)$ is much less so.

Question
Can we improve heuristics for rank classification? Did we loose some of the high-rank curves?

Approach by convolutional neural networks

In our previous work

Kazalicki, Vlah: Ranks of elliptic curves and deep neural networks, Res. number theory 9, 53 (2023)

inspired by the work of He, Lee, Oliver on machine learning invariants of arithmetic curves,
we investigated a deep learning models for rank classification based on convolutional neural networks (CNN).

Approach by convolutional neural networks

In our previous work

Kazalicki, Vlah: Ranks of elliptic curves and deep neural networks, Res. number theory 9, 53 (2023)

inspired by the work of He, Lee, Oliver on machine learning invariants of arithmetic curves,
we investigated a deep learning models for rank classification based on convolutional neural networks (CNN).

These models take as input the conductor of the elliptic curve together with the sequence of normalized a_p-s (i.e. $a_p/\sqrt{p}$) in a fixed range (e.g. $p < 10^k$ za $k = 3, 4, 5$) and predict the rank of the elliptic curve.

Matthews correlation coefficient

- balanced measure of classification quality
  - even for classes of very different sizes
- e.g. for binary classification, MCC is computed using:

$$\mathrm{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP) \cdot (TP + FN) \cdot (TN + EP) \cdot (TN + FN)}}$$

  - TP - number of true positives
  - FN - number of false negatives
  - TN - number of true negatives
  - FP - number of false positives
- MCC lies in the segment $[-1, 1]$
- $\mathrm{MCC} = 1$ only in the case of perfect classification

Rank Classification via Multi-Value Mestre-Nagao Sums

Rank Classification via Multi-Value Mestre-Nagao Sums

How can we classify the rank of an elliptic curve based on the values of the Mestre–Nagao sum for different choices of B?

Rank Classification via Multi-Value Mestre-Nagao Sums

How can we classify the rank of an elliptic curve based on the values of the Mestre–Nagao sum for different choices of B?

For $B \in \{1000, 5000, 10000, 20000, 30000, 40000, 50000, 100000\}$, we propose a classification approach that considers the conductor N of the elliptic curve, alongside selected values of $S_0(B)$ and $S_5(B)$.

Top range test

First, we trained fully connected neural network models on elliptic curves with conductors in the range $[1, 10^8]$ (4,875,676 curves) and tested them on curves with conductors in the range $[10^8, 10^9]$ (12,512,753 curves). We used Balakrishnan et al. (2016) dataset.

Top range test

First, we trained fully connected neural network models on elliptic curves with conductors in the range $[1, 10^8]$ (4,875,676 curves) and tested them on curves with conductors in the range $[10^8, 10^9]$ (12,512,753 curves). We used Balakrishnan et al. (2016) dataset.

B	S_0 and S_5	S_0	S_5
All	0.795	0.737	0.608
$10^3, 10^4, 10^5$	0.786	0.726	0.609
$10^3, 10^5$	**0.784**	0.709	**0.595**
$10^4, 10^5$	0.740	0.705	0.562
$5 \cdot 10^4, 10^5$	0.710	0.685	0.537
10^5	0.699	**0.686**	**0.527**
10^4	0.553	0.537	0.384
10^3	0.330	0.309	0.277

Figure 2: MCC for different model predictions for top range test. Tested with curves in the conductor range $[10^8 + 1, 10^9]$.

Confusion matrices

rank	0	1	2	3	4
0	23.674	6.497	0.336	0.000	0.000
1	3.854	41.410	1.578	0.003	0.003
2	0.474	0.521	18.352	0.003	0.000
3	0.000	0.001	0.005	3.134	0.000
4	0.000	0.000	0.000	0.000	0.153

Figure 3: Confusion matrix for NN predictions with S_0, S_5, and all B-s.

rank	0	1	2	3	4
0	6.607	17.907	5.977	0.016	0.000
1	4.403	27.587	14.301	0.558	0.000
2	0.000	2.602	15.764	0.984	0.000
3	0.000	0.000	0.462	2.677	0.000
4	0.000	0.000	0.000	0.043	0.110

Figure 4: Confusion matrix for NN predictions with S_0 only, $B = 1000$.

Uniform test range

Next, we trained and tested fully connected neural network models on elliptic curves with conductors in the range $[1, 10^9]$, using a dataset of $17,388,429$ curves split into 20% for validation, 20% for testing, and 60% for training (Balakrishnan et al. (2016) dataset).

Uniform test range

Next, we trained and tested fully connected neural network models on elliptic curves with conductors in the range $[1, 10^9]$, using a dataset of $17,388,429$ curves split into 20% for validation, 20% for testing, and 60% for training (Balakrishnan et al. (2016) dataset).

B	S_0 and S_5	S_0	S_5
All	0.856	0.787	0.681
$10^3, 10^4, 10^5$	0.855	0.769	0.661
$10^3, 10^5$	**0.835**	0.747	**0.637**
$10^4, 10^5$	0.787	0.739	0.601
$5 \cdot 10^4, 10^5$	0.752	0.729	0.578
10^5	0.733	**0.712**	**0.573**
10^4	0.596	0.573	0.422
10^3	0.371	0.359	0.295

Figure 5: MCC for different model predictions for uniform test

Confusion matrices

Rank	0	1	2	3	4
0	26.035	4.104	0.345	0.000	0.000
1	3.443	42.346	1.237	0.002	0.000
2	0.061	0.147	19.148	0.002	0.000
3	0.000	0.000	0.000	2.998	0.000
4	0.000	0.000	0.000	0.000	0.131

Figure 6: Confusion matrix for NN predictions with S_0, S_5, and all B-s.

Rank	0	1	2	3	4
0	7.329	17.556	5.559	0.040	0.000
1	3.463	30.043	12.592	0.931	0.000
2	0.000	2.054	15.894	1.404	0.007
3	0.000	0.000	0.126	2.846	0.027
4	0.000	0.000	0.000	0.001	0.130

Figure 7: Confusion matrix for NN predictions with S_0 only, $B = 1000$.

Simplified two-sum model

Question

How can we determine the rank based on the two sums $S_0(100\,000)$ *and* $S_0(1000)$*? What criteria should we use to decide on cutoffs?*

Simplified two-sum model

Question
How can we determine the rank based on the two sums $S_0(100\,000)$ *and* $S_0(1000)$*? What criteria should we use to decide on cutoffs?*

We can attempt to model cutoff areas using rectangles...

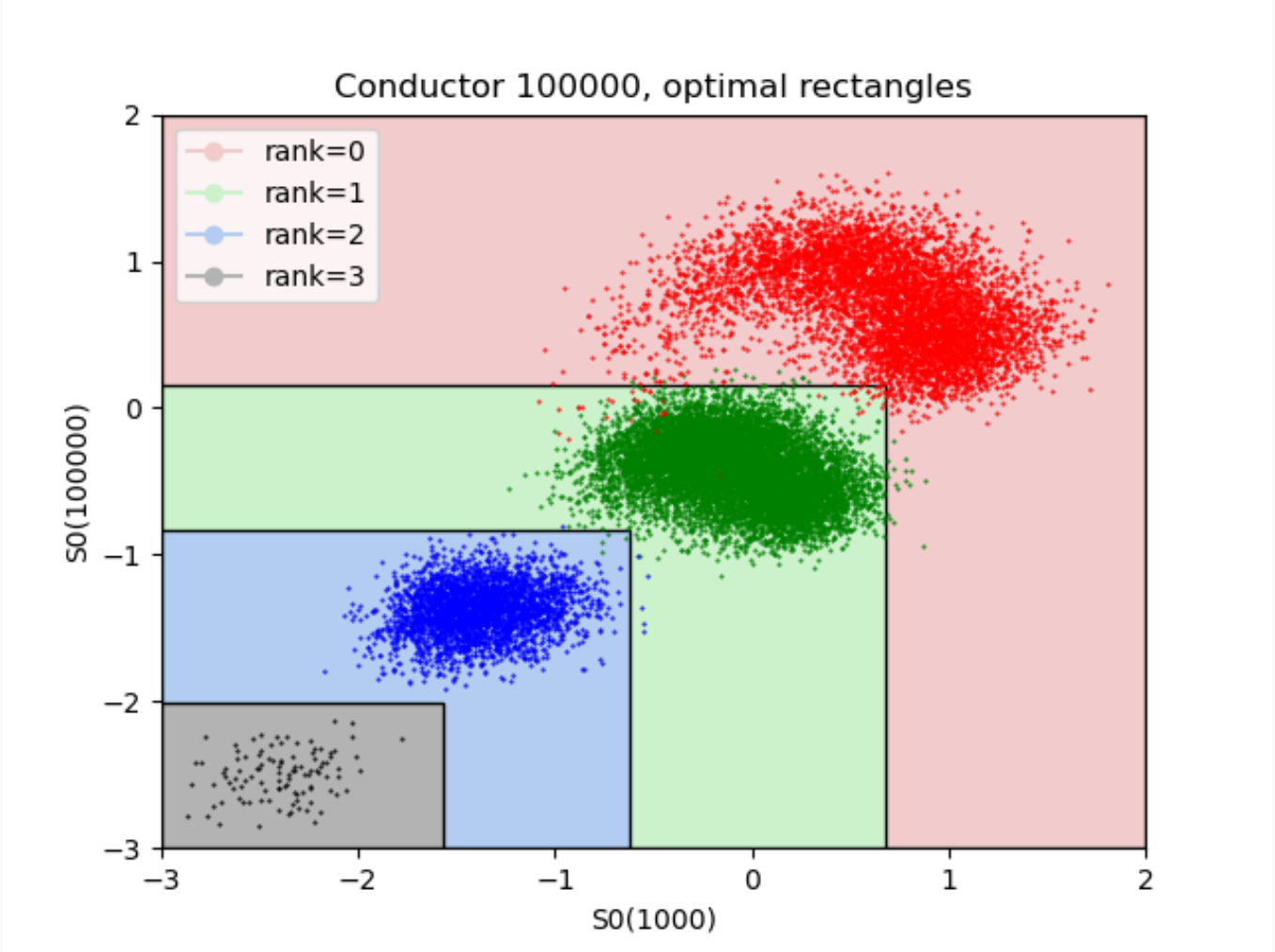


Figure 8: MCC = 0.9921 for classification based on rectangles.
Plotted all curves for conductors in $[100\,000, 150\,000]$.
The MCC for classification within this conductor range based solely on $S_0(100,000)$ is 0.9826.

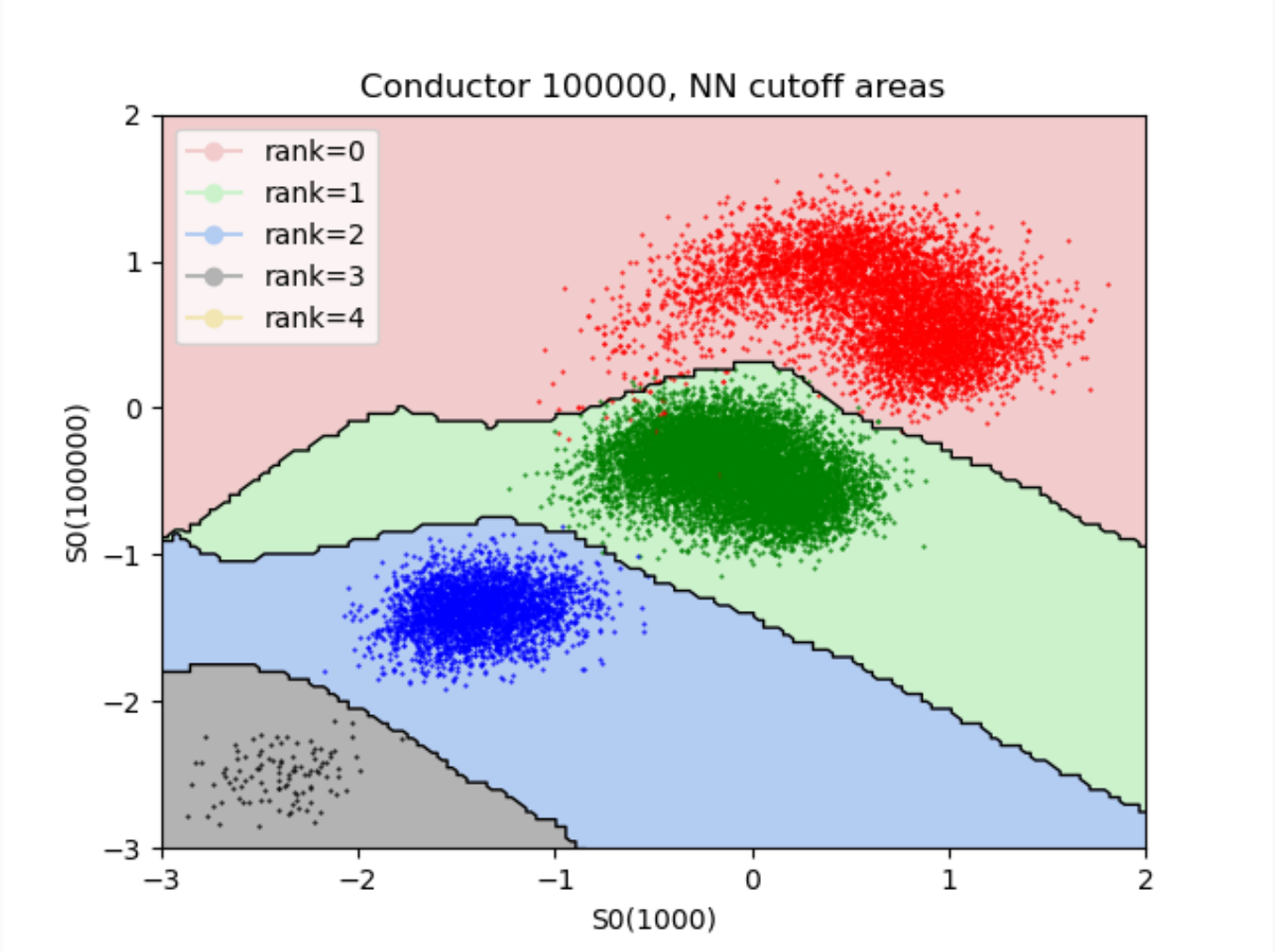


Figure 9: Cutoff areas for neural network model base on $S_0(1000)$ and $S_0(100\,000)$.
Plotted all curves for conductors in $[100\,000, 150\,000]$.
The MCC for classification using the NN model increased to 0.9962.

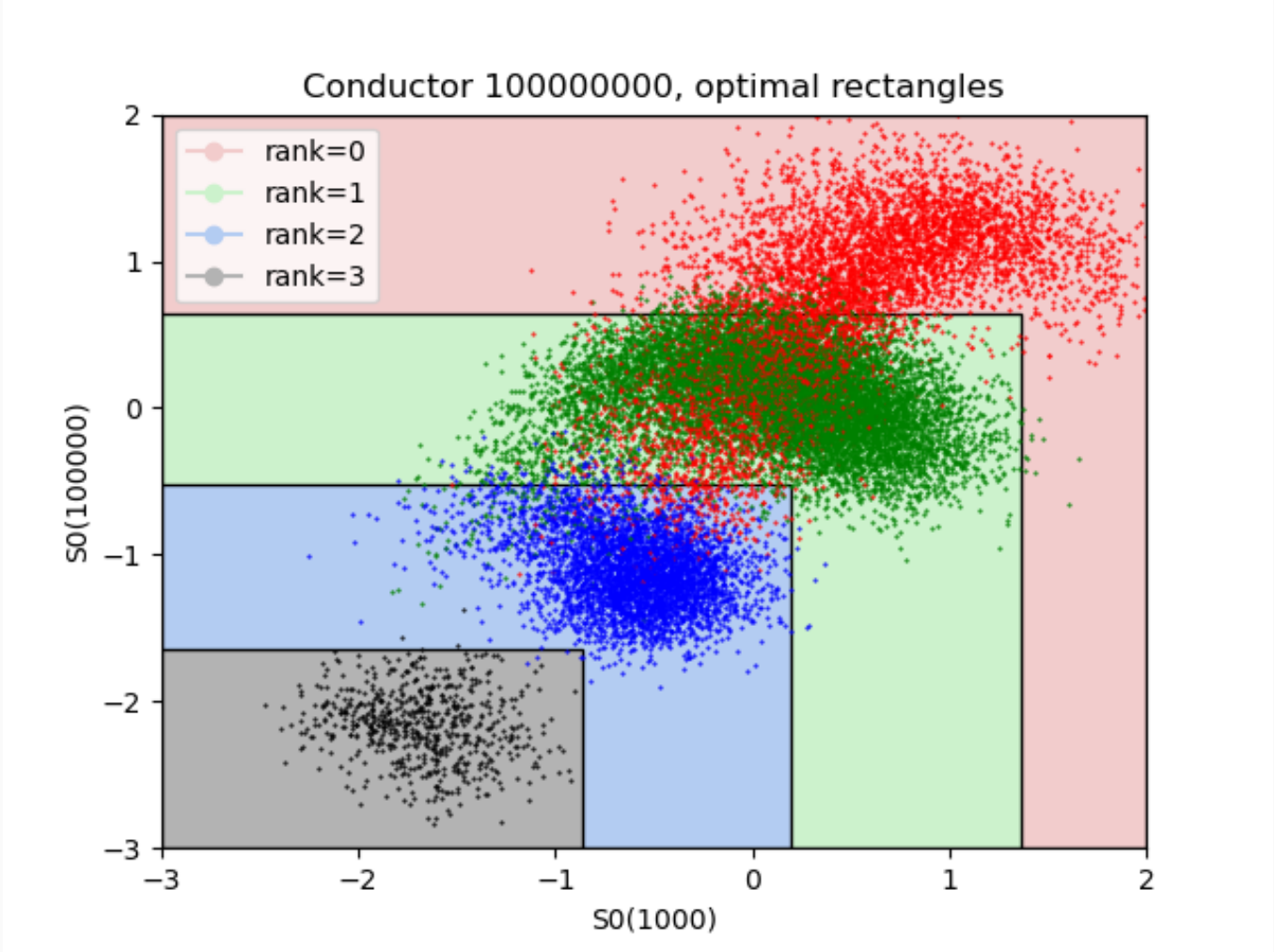


Figure 10: MCC=0.7649 for classification based on rectangles.
Plotted all curves for conductors in $[100\,000\,000, 100\,750\,000]$.
The MCC for classification within this conductor range based solely on $S_0(100,000)$ is 0.7530.

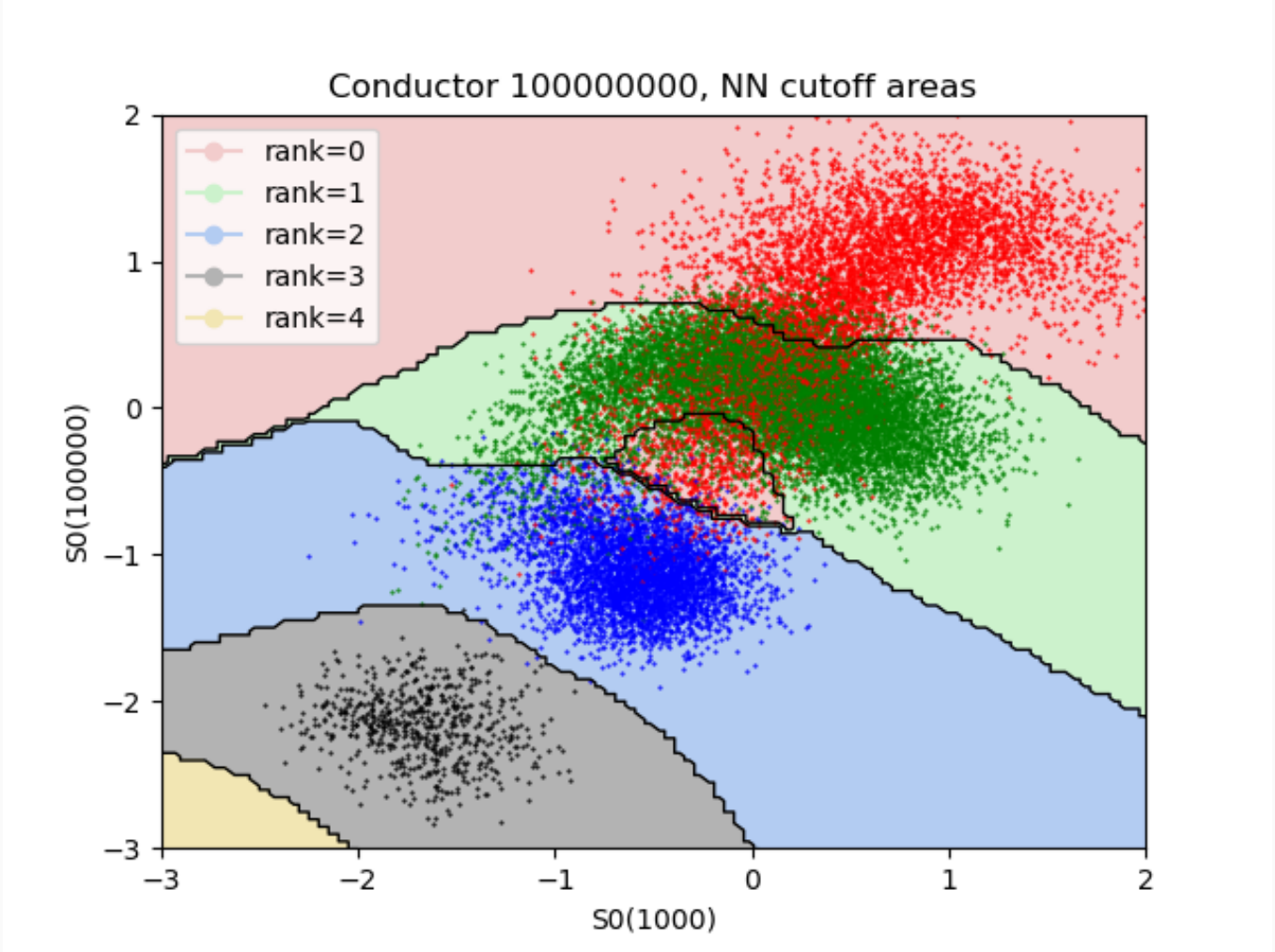


Figure 11: Cutoff areas for neural network model base on $S_0(1000)$ and $S_0(100\,000)$.
Plotted all curves for conductors in $[100\,000\,000, 100\,750\,000]$.
The MCC for classification using the NN model increased to 0.7745

Learning optimal Mestre-Nagao sums

Generalizing S_0

We consider sums of the form $\sum_{p<B} c_p \frac{a_p}{\sqrt{p}}$, where $c_p \in \mathbb{R}$.

Question

Can we improve rank classification of S_0 by replacing the coefficients $c_p = \frac{\log p}{\sqrt{p}}$ with a different function that depends only on p?

Generalizing S_0

We consider sums of the form $\sum_{p<B} c_p \frac{a_p}{\sqrt{p}}$, where $c_p \in \mathbb{R}$.

Question
Can we improve rank classification of S_0 by replacing the coefficients $c_p = \frac{\log p}{\sqrt{p}}$ with a different function that depends only on p?

- Fix $B = 100\,000$.
- Model each coefficient c_p using a fully connected neural network with 4 hidden layers and 128 neurons in each layer.
- The input to the c_p network consists of (normalized) index of p.
- The output is learned c_p coefficient.
- After computing sum with c_p coefficients, we proceed with fully connected neural network responsible for classification, as before.

Results

- The MCC of the network trained on a dataset of elliptic curves with conductors in the range $[1, 10^9]$, using a uniformly chosen test set, is MCC $= 0.7283$.
- This represents a small improvement over the MCC $= 0.712$ obtained from classification based on the S_0 sum.

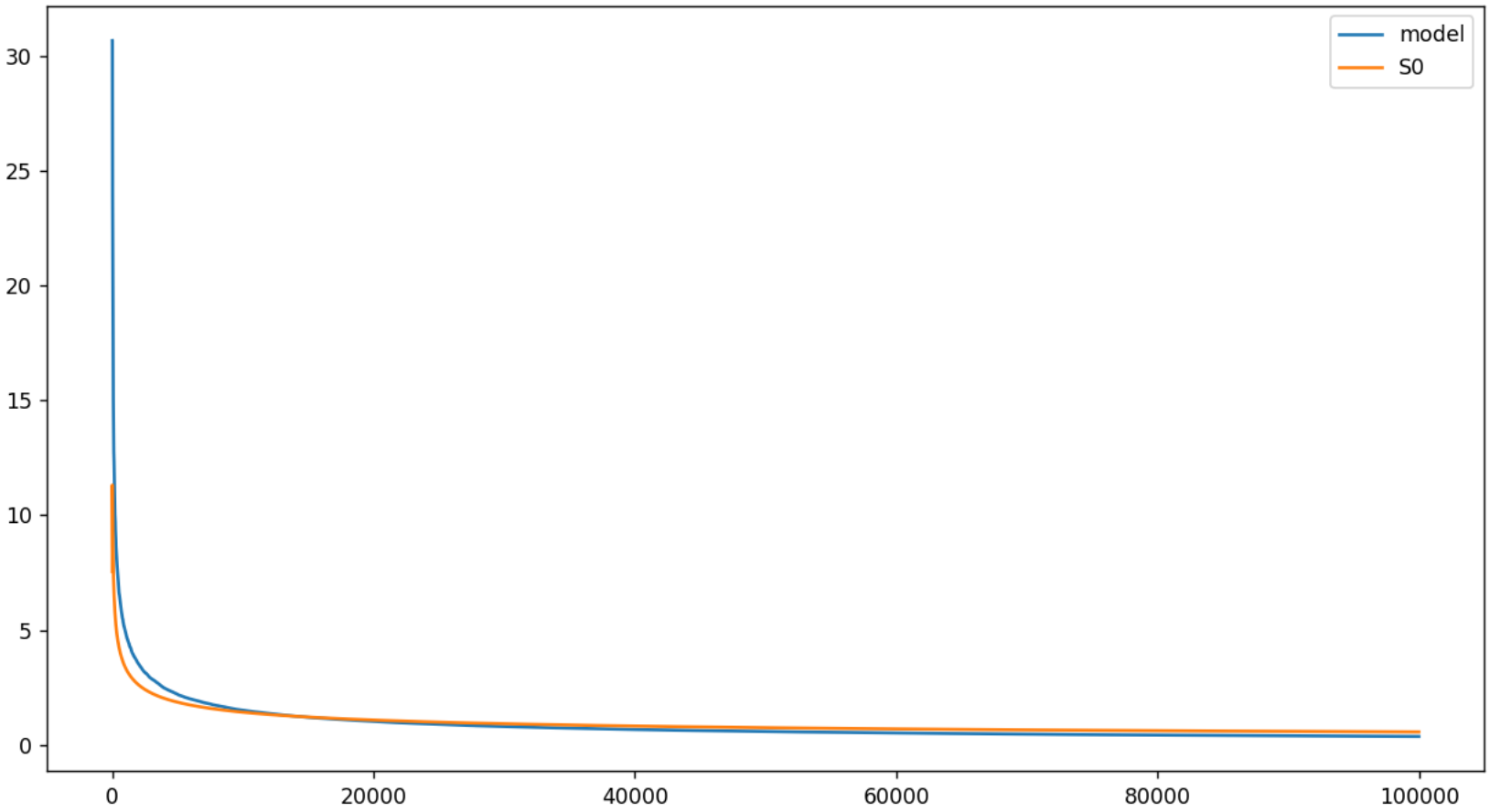
Results

- The MCC of the network trained on a dataset of elliptic curves with conductors in the range $[1, 10^9]$, using a uniformly chosen test set, is MCC $= 0.7283$.
- This represents a small improvement over the MCC $= 0.712$ obtained from classification based on the S_0 sum.

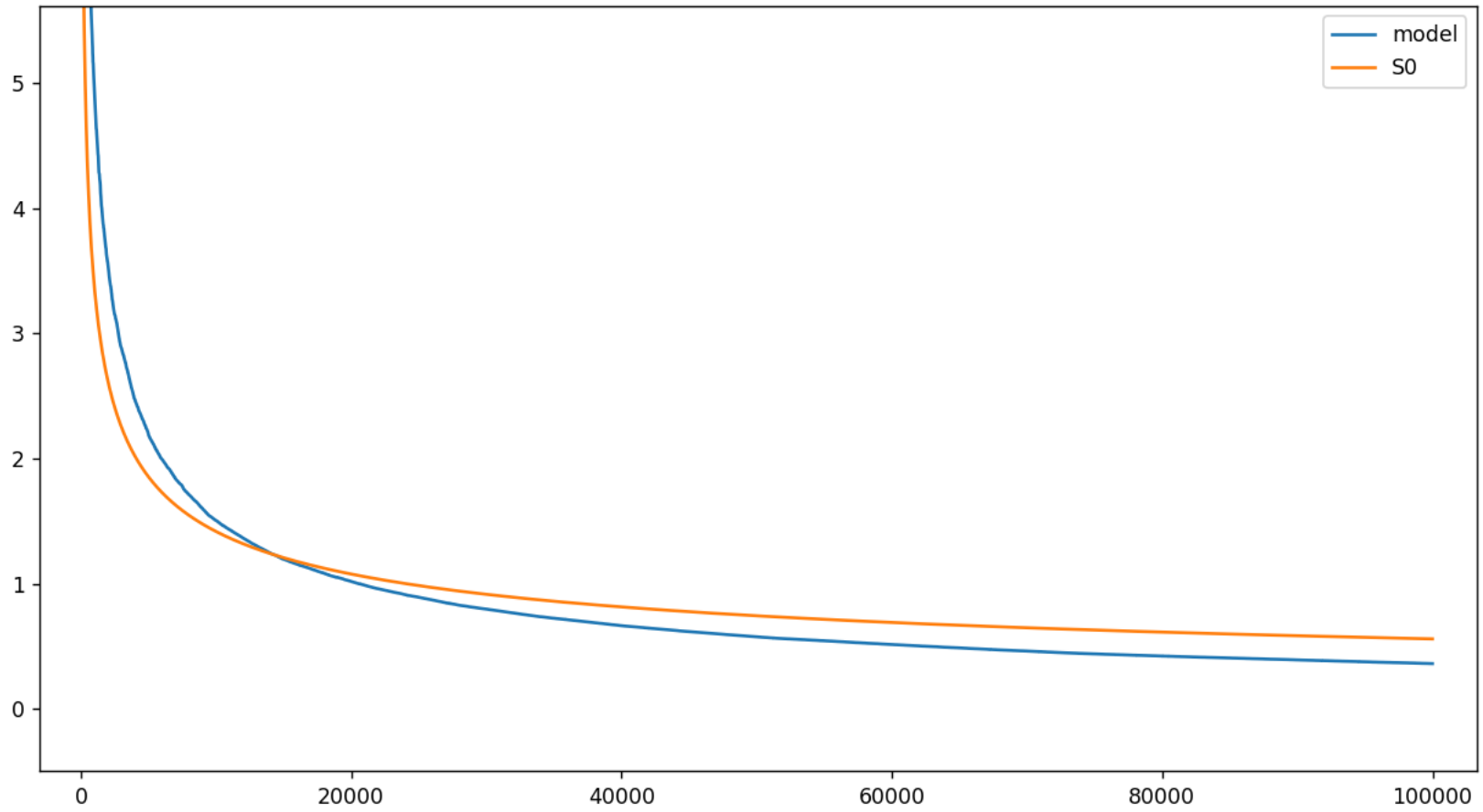
	0	1	2	3	4	5
0	20.4541	9.2744	0.7971	0.0000	0.0000	0.0000
1	5.3150	39.4822	2.1722	0.0021	0.0000	0.0000
2	0.1514	0.0005	19.1926	0.0081	0.0000	0.0000
3	0.0000	0.0000	0.0002	3.0194	0.0000	0.0000
4	0.0000	0.0000	0.0000	0.0000	0.1299	0.0000
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0009

Table 1: Confusion matrix

Comparison with S_0



Comparison with S_0



Model with conductor-dependent coefficients

Similar to the previous model, but now we also provide the (log-normalized) conductor as an input to the fully connected neural network, which now has 256 neurons in each hidden layer, to compute the coefficients c_p.

Results

- The MCC of the network trained on a dataset of elliptic curves with conductors in the range $[1, 10^9]$, using a uniformly chosen test set, is MCC $= \mathbf{0.7322}$.
- This represents a small improvement over the MCC $= 0.7283$ obtained from the previous classifier.

Results

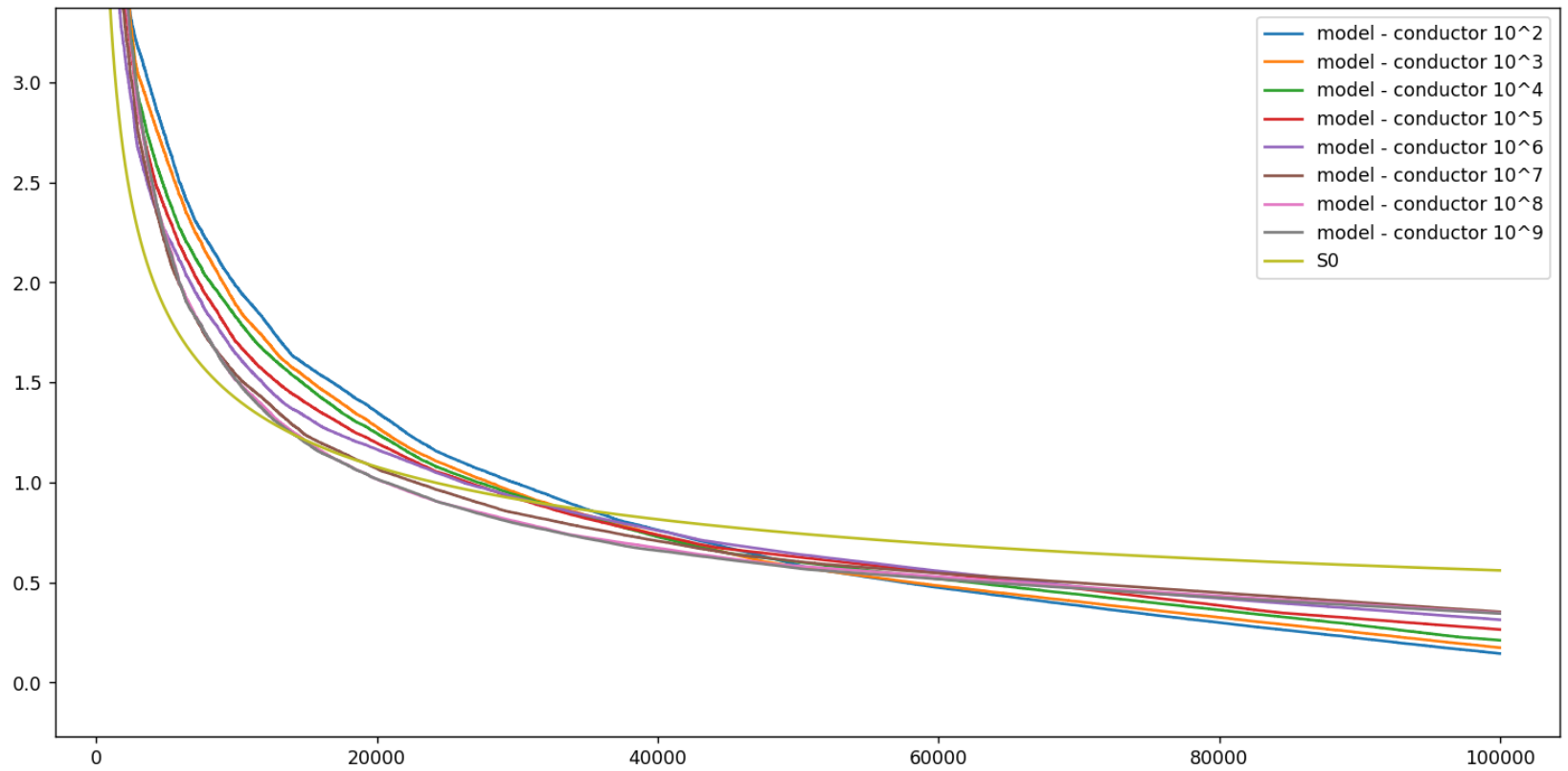
- The MCC of the network trained on a dataset of elliptic curves with conductors in the range $[1, 10^9]$, using a uniformly chosen test set, is MCC $= \mathbf{0.7322}$.
- This represents a small improvement over the MCC $= 0.7283$ obtained from the previous classifier.

	0	1	2	3	4	5
0	20.3458	9.2371	0.9427	0.0000	0.0000	0.0000
1	4.9485	39.8230	2.1971	0.0029	0.0000	0.0000
2	0.1512	0.0007	19.1919	0.0089	0.0000	0.0000
3	0.0000	0.0000	0.0001	3.0194	0.0000	0.0000
4	0.0000	0.0000	0.0000	0.0000	0.1299	0.0000
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0009

Table 2: Confusion matrix for conductor-dependent network

Optimal coefficients for different conductor ranges

This conductor-dependant model allows us to observe the optimal coefficients for each conductor range.



Question
Can one identify the optimal coefficients?

Question
Can one identify the optimal coefficients?

Question
How does the model behave on high-rank curves?

Thank you for your attention!

Thank you for your attention!

More details:

Bujanović, Kazalicki, Vlah: Improving elliptic curve rank classification using multi-value and learned Mestre–Nagao sums, accepted for publication in IJDSMS