Supplementary Materials

February 12, 2026

1 HFV Generation

1.1 Technical Details

The data generation pipeline is implemented using snakemake [4], Python and libraries including KMC [1], `quickmers` [2], `bbhash` [3] and `pybloomfilter3` [5].

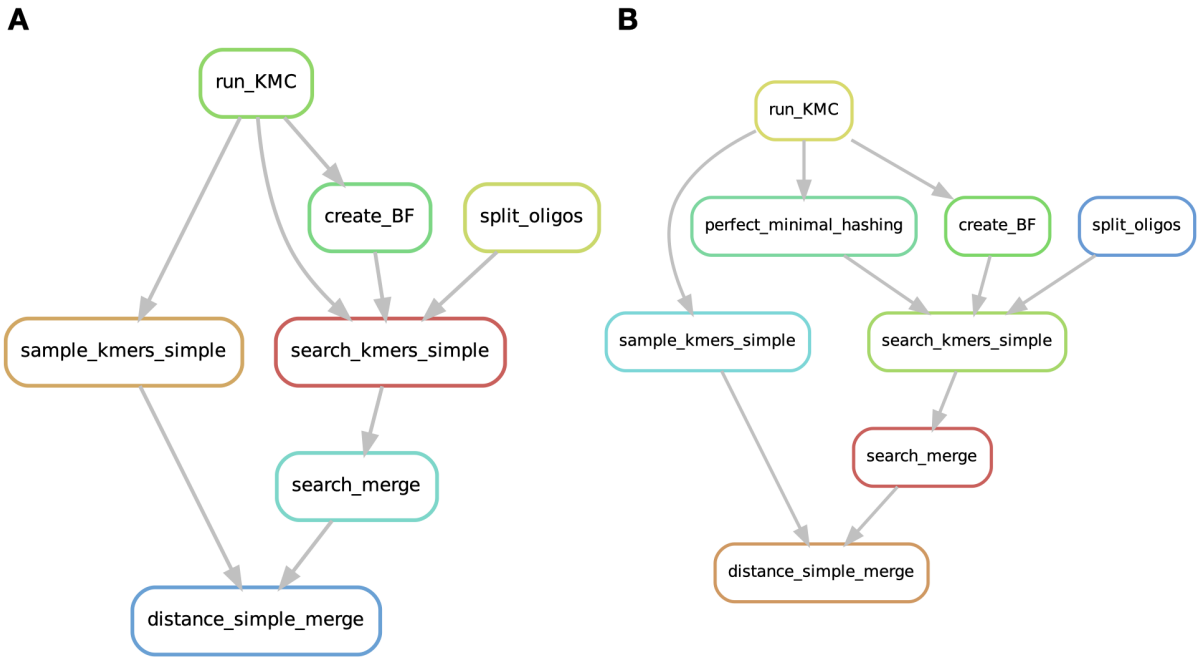
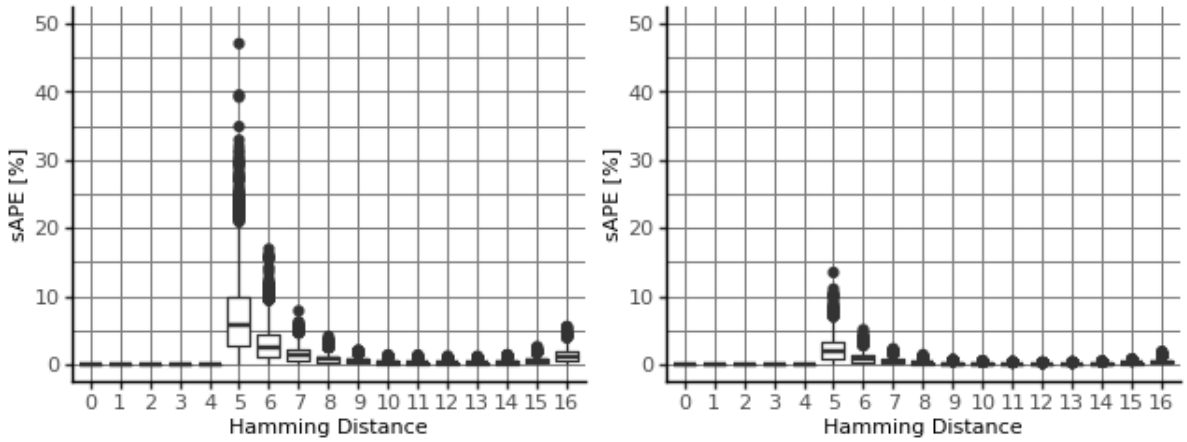


Figure S1: Snakemake DAGs for generating HFVs using Linear Search (A) and MPH (B).

1.2 Error Rates of resulting Hamming-based HFVs using Linear Search

Figure S3 shows the accuracy of HFVs for Hamming distances generated with 100 000 and 1 000 000 samples, respectively, compared to exact HFVs for 2000 query sites. It can be shown that an increase

of a magnitude for the number of samples drawn can significantly improve the accuracy of the resulting HFVs. For Hamming distances of up to 4, an exact search is performed indicated by the sAPEs of 0.0%.



(a) (b)

Figure S2: Symmetric Absolute Percentage Errors (sAPE) of HFVs resulting from the hybrid sampling and search approach compared to exactly computed HFVs per frequency bin using 100 000 samples (a) and 1 000 000 samples (b).

1.3 Error Rates of resulting Hamming-based HFVs using Minimal Perfect Hashing

Figure S3 shows the accuracy of HFVs for Hamming distances generated with 100 000 samples, compared to exact HFVs for 2000 query sites. It can be shown that an increase of a magnitude for the number of sampled sites can significantly improve the accuracy of the resulting HFVs. For Hamming distances of up to 4 an exact search is performed indicated by the sAPE of 0.0%.

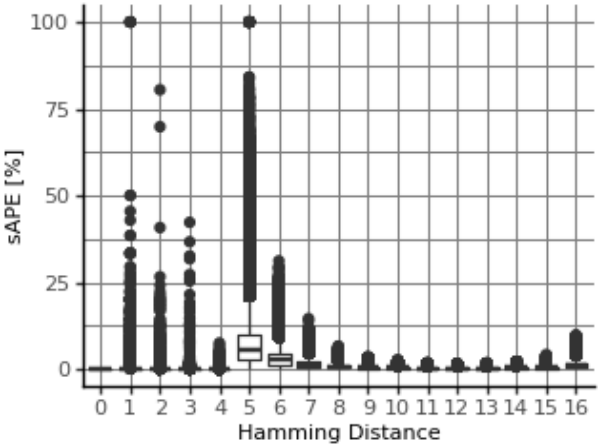


Figure S3: sAPE of HFVs resulting from the hybrid sampling and search approach using MPH compared to exactly computed HFVs per frequency bin using 100 000 samples.

1.4 Data set statistics

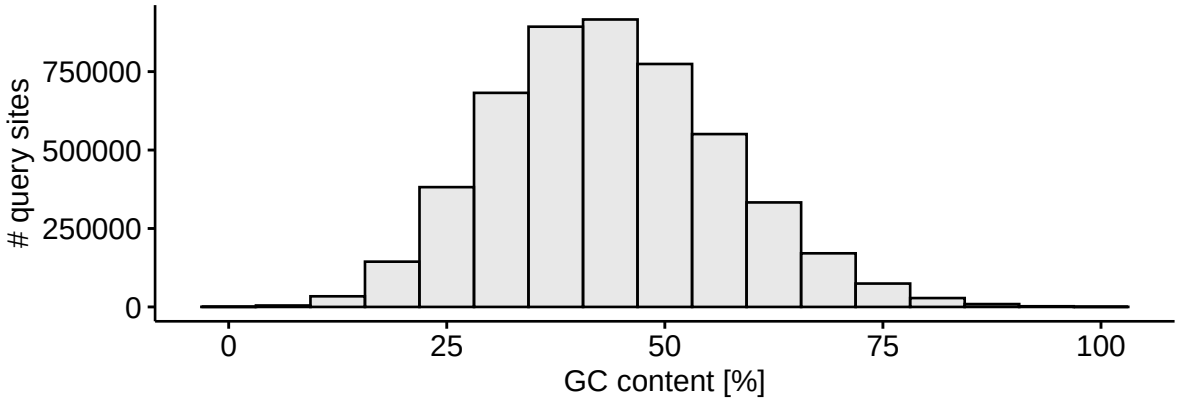


Figure S4: GC content distribution of the query sites X.

2 Model Architectures

2.1 KD

Table S1: Architecture of KD model.

Layer	**In Channels**	**Out Channels**	**Filter or dropout**
2D Convolution	1	64	3×3
2D Convolution	64	64	3×3
Max Pool	*	*	*
Drop Out	*	*	0.2
Fully connected	1024	1024	*
Fully connected	1024	256	*
Fully connected	256	128	*

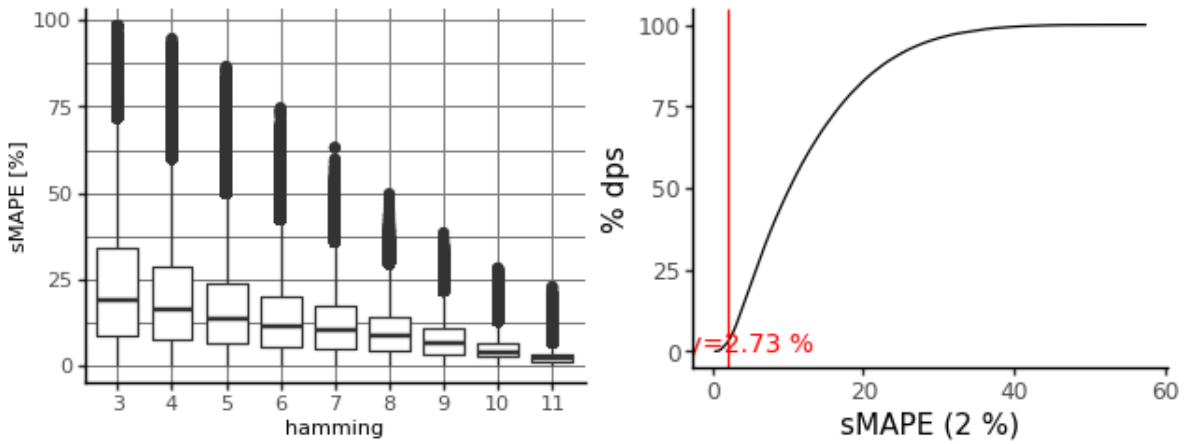
2.2 BUT

Table S2: Architecture of BUT model.

Layer	**In Channels**	**Out Channels**	**Kernel size or dropout**
2D Convolution	1	512	4×2
Drop Out	*	*	0.2
1D Convolution	512	256	3
1D Convolution	256	64	1
Drop Out	*	*	0.2
Fully connected	*	256	*
Fully connected	256	128	*
Fully connected	128	64	*

2.3 Median Regression Baseline

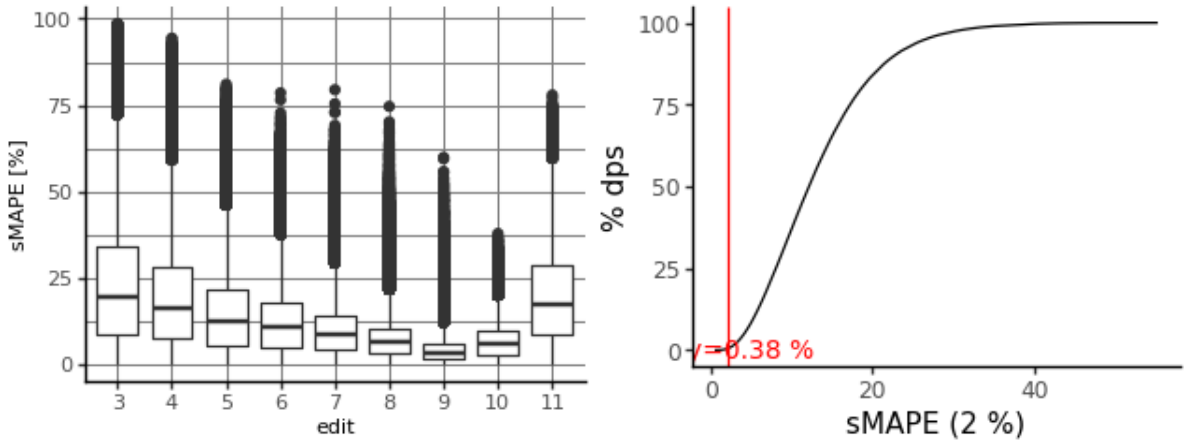
Specifically, the Hamming HFVs display a significant homogeneity across 5 million generated HFVs. Therefore, a simple median regression model is implemented to evaluate the validity of using neural networks for these predictions. Figure S5 shows the results for the median regression model. While the sAPE per bin in Figure S5a show that the predictions for larger distances, e.g. Hamming distance=11, have low errors for predicting the median. The baseline models performs bad on low to medium Hamming distances. Only about 2.7 % of the predictions have a mean sAPE of less than 2 % compared to 60 % for the KD model.



(a) (b)

Figure S5: Results for the median regression baseline model for Hamming HFVs. (a) sAPE per bin for the median regression model. (b) Percentage of data points in the test set that have a mean sAPE less than the given threshold (x-axis).

The median regression baseline was also performed for predicting Edit HFVs (cf. Figure S6). Edit HFVs show a higher heterogeneity compared to Hamming HFVs. Therefore, the median regression baseline performs significantly worse with 0.4 % of the predictions having a mean sAPE of less than 2 %. The KD model achieves 55 % of the predictions with a mean sAPE of less than 2 %.



(a) (b)

Figure S6: Results for the median regression baseline model for Edit HFVs. (a) sAPE per bin for the median regression model. (b) Percentage of data points in the test set that have a mean sAPE less than the given threshold (x-axis).

3 Cross-Fold Validations

3.1 Single point predictions

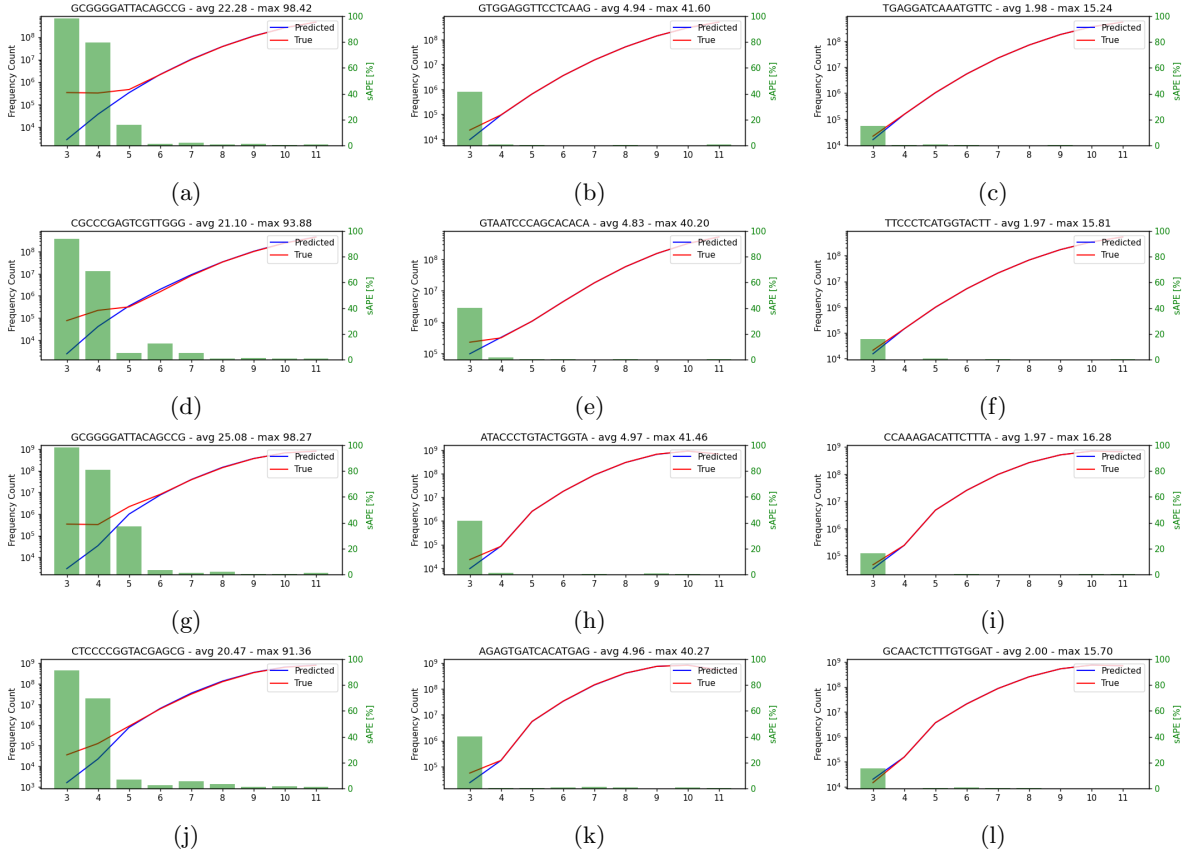
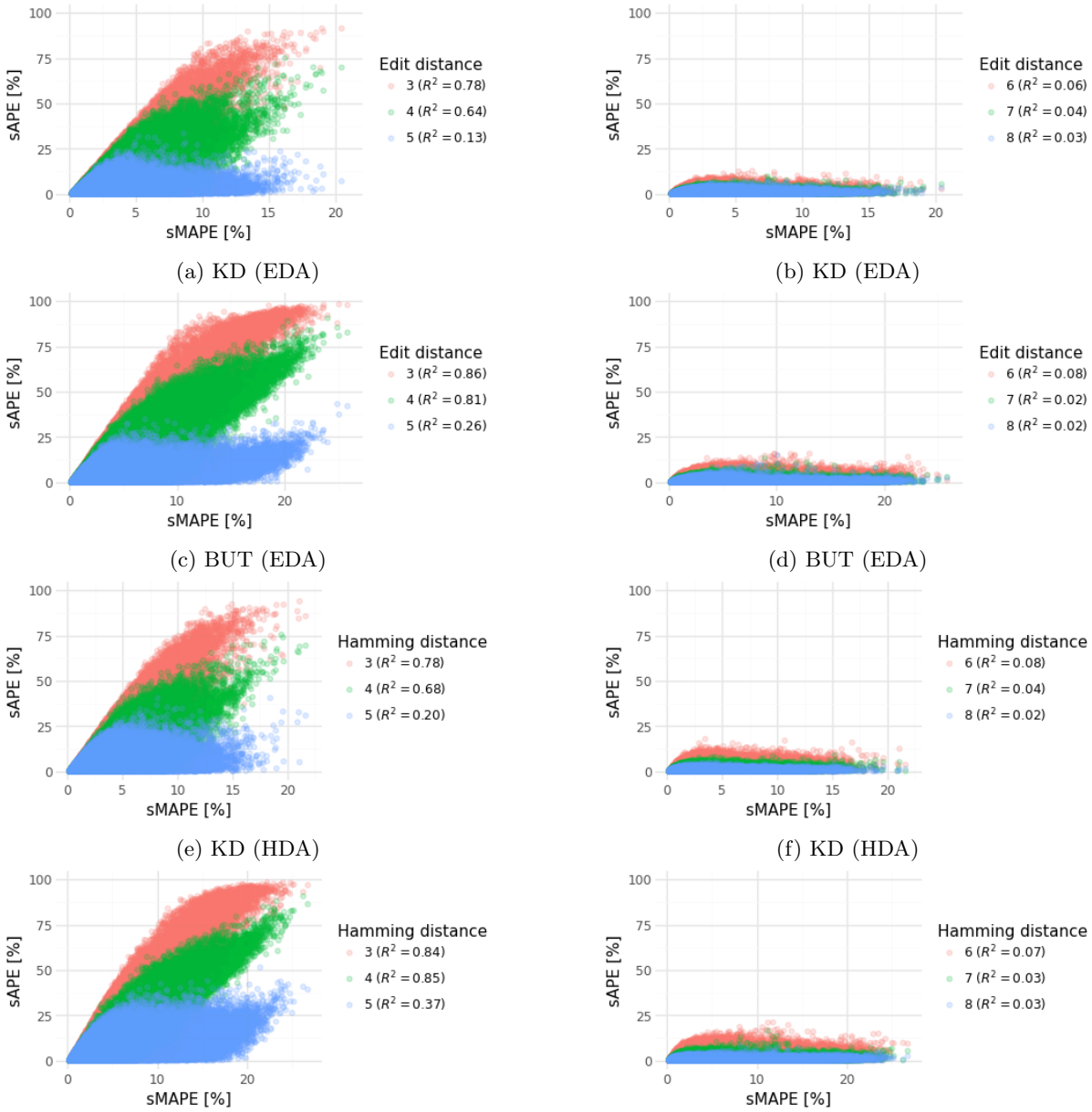


Figure S7: Single-point HFV predictions for Hamming HFVs using BUT (a,b,c) and KD (d,e,f), and for Edit HFVs using BUT (g,h,i) and KD (j,k,l). First sequence prediction for each row shows the worst prediction overall, second shows the prediction with the highest symmetric APE error with an average sMAPE $< 5\,\%$ and the last shows the HFV prediction with highest sAPE and an average sMAPE $< 2\,\%$.



(a) KD (EDA)

(b) KD (EDA)

(c) BUT (EDA)

(d) BUT (EDA)

(e) KD (HDA)

(f) KD (HDA)

(g) BUT (HDA)

(h) BUT (HDA)

Figure S8: Correlation of sAPEs in distance specific bins with the sMAPE per data point.

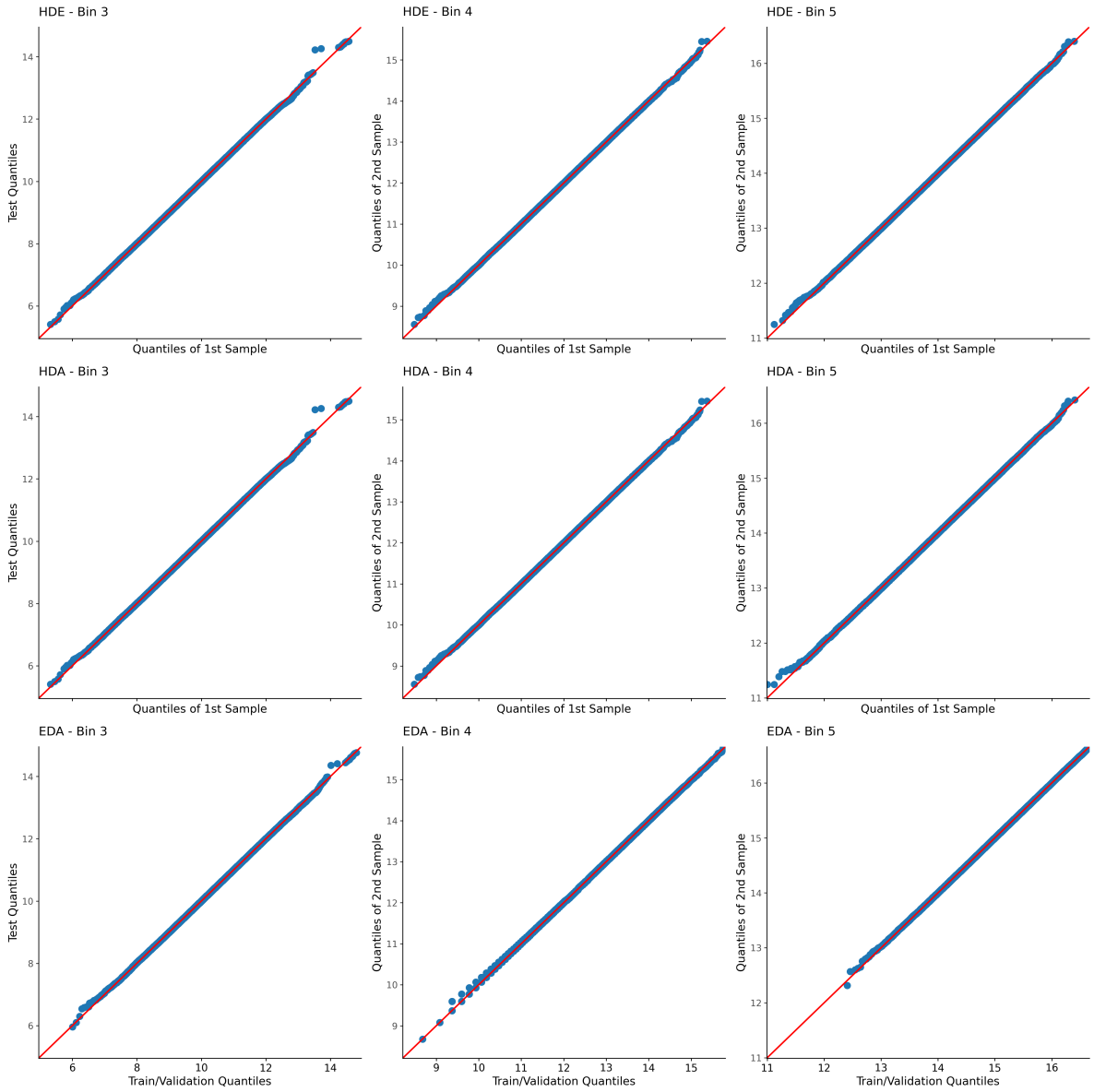


Figure S9: QQ-plots for frequency bins 3 to 5 for HDE, HDA and EDA.

3.2 Prediction outliers

Table S3: Percentages of outliers per bin prediction for Hamming HFV and Edit HFV for model architectures KD and BUT, respectively.

	Hamming HFVs		**EDA**	
Bin/distance	**BUT**	**KD**	**BUT**	**KD**
3	8.1	7.4	7.9	7.7
4	8.4	7.0	8.0	7.1
5	5.1	3.8	4.6	4.2
6	3.2	3.0	3.7	3.0
7	2.6	2.4	3.1	2.3
8	2.2	2.0	3.1	2.1
9	1.6	2.0	2.8	2.0
10	1.9	2.1	1.3	2.0
11	0.8	2.3	1.9	2.4

4 GenMap

Listing 1: Genmap command

```
THREADS=16
K=16
E=2 # changed accordingly

genmap map -K $K -E $E -I indices/GCA_000001405_skew -O output/GCA_000001405_skew/
    GCA_000001405.15_GRCh38_no_alt_analysis_set_'$K'_'$E'_'$THREADS'.genmap.txt -t -fs -
    r -m -nc -T $THREADS
```

Abbreviations

DL deep learning

HFV Hit Frequency Vector

HSS Hybrid Search and Sampling

CNN convolutional neural network

MOR multi-output regression

MPH minimal perfect hashing

MSE mean squared error

MAE mean absolute error

sAPE symmetric absolute percentage error

sMAPE symmetric mean absolute percentage error

IQR interquartile range

References

[1] Sebastian Deorowicz et al. "KMC 2: fast and resource-frugal k-mer counting". In: *Bioinformatics* 31.10 (Jan. 2015), pp. 1569–1576. DOI: 10.1093/bioinformatics/btv022. URL: http://dx.doi.org/10.1093/bioinformatics/btv022.

[2] Kian Jalilian. *Quickmers*. Dec. 2025. URL: https://github.com/Schlieplab/quickmers.

[3] Antoine Limasset et al. "Fast and Scalable Minimal Perfect Hashing for Massive Key Sets". en. In: vol. 75. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017, 25:1–25:16. DOI: 10.4230/LIPICS.SEA.2017.25. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SEA.2017.25.

[4] F Mölder et al. "Sustainable data analysis with Snakemake [version 3; peer review: 2 approved]". In: *F1000Research* 10.33 (2025). DOI: 10.12688/f1000research.29032.3.

[5] Prashant Sinha and Vytautas Mizgiris. *Pybloomfiltermmap3*. URL: https://github.com/prashnts/pybloomfiltermmap3.