ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Scaling up graph-based classifiers with a divide and conquer approach

Caius V. C. M. Souza[1], Rafael L. Almeida[2],
Frederico G. F. Coelho[1] and Antônio P. Braga[1] *

1- Universidade Federal de Minas Gerais - UFMG
Belo Horizonte, Brazil

2- University of Eastern Finland - UEF
Kuopio, Finland

Abstract. Traditional techniques, such as SVMs, may become impractical for real-world applications. This case study introduces a divide-and-conquer ensemble framework, namely Gabriel Graph Network Ensemble, applied to ChipClass, a graph-based SVM. By training on independent data partitions and then aggregating predictions, GGNE achieves a 1500x speed up while preserving comparable AUC across 23 datasets. These results demonstrate that our ensemble decomposition can achieve critical acceleration without risking performance, offering a feasible approach for classifying massive volumes of data.

1 Introduction

Classical models are often impractical for real-world applications that require handling the emerging large data sets because of their limited scalability, flexibility, and need for manual feature engineering, prompting practitioners to opt for more costly solutions like deep learning, which also have their own limitations [1].

Support Vector Machines (SVMs) [2], in particular, are known for their poor scalability, with quadratic or cubic complexity, yet they remain competitive because of their capability to model complex nonlinear relationships, handle imbalanced data, and rely on solid theoretical foundations [3]. ChipClass [4], a support vector-based classifier, utilizes Gabriel graphs for parameter selection, eliminating the need for tuning, yet it faces the challenge of cubic complexity in graph construction.

To tackle these scalability problems, we propose a divide-and-conquer strategy for ChipClass known as the Gabriel Graph Network Ensemble (GGNE). This approach can speed up the training process while maintaining the original predictive performance.

*The authors from UFMG would like to thank CAPES, CNPq, FAPEMIG and the PPGEE for the support given to this work.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

2 Literature Review

2.1 Gabriel Graph and ChipClass

Consider a dataset represented by N samples in d dimensions, $X \in \mathbb{R}^{N \times d}$. A *Gabriel graph* [5] is a planar geometric graph constructed over these samples: an edge exists between two samples if the closed hypersphere with the segment connecting them as diameter contains no other sample. Formally, nodes u and v are connected if

$$\delta^2(u, v) < \delta^2(u, i) + \delta^2(v, i) \quad \text{for all } i \neq u, v,$$

where $\delta(\cdot, \cdot)$ denotes Euclidean distance. Gabriel graphs capture geometric and structural information effectively, with applications from networking topology analysis [6] to molecular modeling [7]. However, constructing them involves cubic complexity based on the number of samples.

ChipClass [4] uses Gabriel graphs to form Structural Support Vectors (SSVs), replacing traditional SVM optimization. It links nearby samples from opposite classes, builds local hyperplanes between them, and combines their outputs using distance-based weights. The method is adaptive, interpretable, and parameter-free, but remains computationally expensive due to graph construction.

2.2 Rvote and Ivote: Ensemble Learning for Large-Scale Data

Ensemble methods like Random Forests [8] have been further developed with algorithms like Random voting (Rvote) and Importance voting (Ivote) [9] to cater to large-scale datasets that exceed the capacity of system memory. In Ivote, the dataset is divided into smaller subsets and independent classifiers are trained on each subset. Their predictions are combined via voting for parallel and memory-efficient training. On the other hand, in Rvote, subsets are randomly sampled and predictions depend on majority voting.

3 Methodology

3.1 Gabriel Graph Network Ensemble

The *Gabriel Graph Network Ensemble* (GGNE)[1] utilizes a non-iterative divide-and-conquer strategy influenced by Breiman's RVote. It segments the dataset into separate subsets through sampling without replacement. Although IVote often outperforms RVote, we chose RVote to retain the original Gabriel graph topology, which is crucial for utilizing its structural advantages.

In GGNE, *ChipClass* functions as the weak learner, cutting down on training time and memory overhead for large datasets by allowing separate, parallel training of submodels. The dataset is randomly divided into K subsets of τ samples each. A Gabriel graph and ChipClass model are constructed for each

[1] Code available at: https://github.com/litc-ufmg/ggne

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

subset. Final predictions are derived through majority voting across all weak learners.

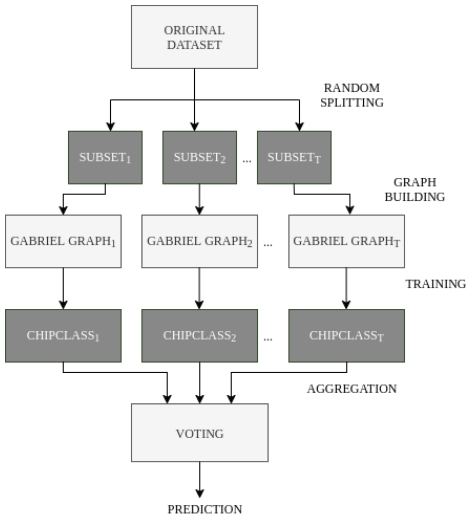

Fig. 1: Overview of the GGNE training and classification process.

We parameterize the subset size using a hyperparameter $\alpha \in (0, 0.5]$, which defines the proportion of the total number of samples included in each subset, i.e., $\tau = \alpha \cdot N$, where N is the total number of samples.

The following pseudocodes describe the training and prediction procedures of GGNE.

Algorithm 1 GGNE Ensemble Training

Input: Training data $\mathbf{D} = \{(X, y)\}$, proportion parameter $\alpha \in (0, 0.5]$.
Output: A list of trained models $\mathcal{M}$.

$N \leftarrow |D|$ ▷ Total number of samples
$\tau \leftarrow \lfloor \alpha \cdot N \rfloor$ ▷ Subset size per model
$K \leftarrow \max(2, \min(\lfloor N/\tau \rfloor, \lfloor N/2 \rfloor))$ ▷ Number of submodels
$\mathcal{M} \leftarrow \emptyset$
Generate K stratified training folds $\mathcal{F}$ from D.
for each index set I in $\mathcal{F}$ **do**
  $D_{aux} \leftarrow$ samples from D at indices I.
  $D_{subset} \leftarrow$ random subset of D_{aux} of size τ (without replacement).
  Train a new base model m on D_{subset}.
  Add m to $\mathcal{M}$.
end for
return $\mathcal{M}$.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Algorithm 2 GGNE Ensemble Prediction (Majority Vote)

Input: Trained ensemble $\mathcal{M}$, test sample X_i.
Output: Predicted label y_{pred}.

```
Initialize v ← 0                                  ▷ Vote count
for each model m in M do
    v ← v + m.predict(X_i)                        ▷ Binary prediction (0 or 1)
end for
y_pred ← (v ≥ |M|/2)                              ▷ Majority vote
return y_pred.
```

3.2 Computational Complexity

The main bottleneck in ChipClass is constructing the Gabriel graph, with worst-case complexity $\mathcal{O}(dn^3)$, where n is the number of samples and d the dimensionality [10]. Each pair (x_i, x_j) requires checking whether the disk with diameter (x_i, x_j) contains any other point x_k, yielding $\mathcal{O}(n^2)$ pairs and $\mathcal{O}(n)$ checks per pair. Each d-dimensional check adds a factor $\mathcal{O}(d)$.

GGNE reduces this cost by constructing Gabriel graphs on K smaller subsets, each with $\tau = \alpha N$ samples, $0 < \alpha < 0.5$. The total complexity becomes:

$$\begin{aligned} K \cdot \mathcal{O}(d \cdot \tau^3) &= \frac{N}{\tau} \cdot \mathcal{O}(d \cdot \tau^3) \\ &= \mathcal{O}(d \cdot N \cdot \tau^2) \\ &= \mathcal{O}(d \cdot N^3 \cdot \alpha^2) \ll \mathcal{O}(d \cdot N^3), \end{aligned} \tag{1}$$

demonstrating that the ensemble significantly reduces computational cost compared to building a single graph on the full dataset, as the cost is scaled by the small factor α^2.

4 Experiments and Results

4.1 Experimental Settings

Models and Tuning. GGNE was evaluated against conventional ChipClass, which does not need tuning. Since GGNE has one hyperparameter, α, for the small datasets, it was optimized via 10-fold cross-validation to select the minimum α achieving the best AUC. For the larger datasets, otherwise, we arbitrarily choose a comparative value of $\alpha = 3\%$, since it empirically leverages performance and fit time.

Datasets. Experiments used 13 small UCI datasets (100–1,372 samples, 3–60 features) and 10 larger OpenML datasets (7400–20,634 samples, 4–86 features). All datasets were standardized using only training samples.

Evaluation Protocol. Performance was measured by AUC and efficiency by fit time, using 5-fold cross-validation. Mean and standard deviation are reported.

Implementation. Experiments ran on CPU x86_64 (Intel Cascade Lake), 16 vCPUs, 64 GB RAM, Python 3.12.9, Ubuntu 24.04 (Linux 6.14.0-28-generic). No GPUs were used.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

4.2 Analysis on smaller data sets

Performance comparison in small datasets. Table 1 indicates that GGNE consistently outperforms ChipClass (CC) in terms of computational cost across the smaller sets of data. Although CC obtains marginally higher AUC in a few cases, GGNE equals or surpasses it in most comparisons. The runtime gains, however, are substantial: depending on the dataset, GGNE operates from slightly faster to over two orders of magnitude faster than CC. This pattern highlights GGNE's ability to deliver reliable classification performance while reducing execution time.

Data set	AUC GGNE (α^*)	AUC CC	Time GGNE (ms)	Time CC (ms)	N/Nd	Speed Up
Australian	**0.89 ± 0.06** (4%)	0.87 ± 0.06	**24.71 ± 1.06**	484.73 ± 46.49	690/14	20
Banknote	0.99 ± 0.01 (9%)	**1.00 ± 0.00**	**50.32 ± 0.93**	5079.30 ± 211.59	1372/4	101
Breast Cancer	**0.94 ± 0.04** (9%)	**0.94 ± 0.04**	**25.49 ± 0.44**	602.82 ± 45.77	699/9	24
Climate	**0.89 ± 0.03** (30%)	**0.89 ± 0.04**	**41.96 ± 0.69**	254.10 ± 4.50	540/20	6
Fertility	0.64 ± 0.35 (43%)	**0.65 ± 0.37**	**3.85 ± 0.26**	4.00 ± 0.22	100/9	1
German	0.69 ± 0.05 (4%)	**0.70 ± 0.04**	**37.29 ± 0.81**	2106.55 ± 102.48	1000/20	56
Haberman	**0.61 ± 0.15** (19%)	**0.61 ± 0.09**	**10.17 ± 0.34**	27.12 ± 0.61	306/3	3
ILPD	**0.64 ± 0.07** (19%)	0.63 ± 0.07	**23.12 ± 0.44**	324.91 ± 18.20	583/10	14
Liver	0.66 ± 0.07 (19%)	**0.70 ± 0.12**	**11.57 ± 0.40**	40.46 ± 2.29	345/6	3
Parkinsons	**0.80 ± 0.09** (43%)	0.75 ± 0.12	**7.77 ± 0.29**	11.36 ± 0.42	195/22	1
Pima	**0.75 ± 0.06** (9%)	**0.75 ± 0.06**	**29.30 ± 1.03**	894.09 ± 56.26	768/8	31
Sonar	**0.82 ± 0.10** (9%)	0.76 ± 0.15	**8.60 ± 0.51**	17.27 ± 1.12	208/60	2
Statlog	**0.85 ± 0.08** (9%)	0.83 ± 0.09	**10.04 ± 0.35**	23.58 ± 0.54	270/13	2
Average Rank	**1.31**	1.38	**1.00**	2.00	-	-

Table 1: AUC and time comparison with optimized α across smaller datasets.

4.3 Analysis on larger data sets

Table 2 shows that GGNE maintains strong scalability advantages over CC on large datasets. Although ChipClass achieves slightly higher AUC in 8 out of 10 cases, GGNE matches or exceeds it on Ringnorm and Naticus, keeping its 900–1400x speed up across data. This shows that GGNE achieves massive efficiency improvements on high dimensional and large sample data while preserving competitive accuracy. Moreover, the acceleration factor increases markedly as the number of samples grows, a property that is particularly advantageous for methods intended for large-scale applications.

Dataset	AUC GGNE (3%)	AUC CC	Time GGNE (s)	Time CC (s)	N/Nd	Speed Up
Phishing Websites	0.88 ± 0.01	**0.90 ± 0.01**	**2.04 ± 0.04**	2759.48 ± 188.55	11055/30	1352
California	0.79 ± 0.00	**0.81 ± 0.01**	**8.05 ± 0.45**	9376.8 ± 61.03	20634/8	1165
Heloc	0.69 ± 0.01	**0.70 ± 0.01**	**1.65 ± 0.04**	1977.57 ± 28.08	10000/22	1199
Jm1	0.62 ± 0.01	**0.63 ± 0.02**	**1.81 ± 0.04**	2552.35 ± 125.91	10885/21	1410
Letter	0.80 ± 0.03	**0.90 ± 0.03**	**7.90 ± 0.19**	9575.87 ± 195.56	20000/16	1212
Mozilla4	0.85 ± 0.01	**0.86 ± 0.02**	**4.02 ± 0.04**	4165.88 ± 26.14	15545/4	1036
Ringnorm	**0.52 ± 0.01**	0.51 ± 0.01	**1.04 ± 0.02**	530.42 ± 24.70	7400/20	510
Electrical	0.80 ± 0.01	**0.83 ± 0.01**	**1.77 ± 0.02**	1592.90 ± 133.79	10000/12	900
Magic Gamma	0.75 ± 0.01	**0.79 ± 0.01**	**6.54 ± 0.11**	9661.88 ± 284.79	19020/10	1477
Naticus	**0.89 ± 0.01**	0.89 ± 0.02	**1.22 ± 0.02**	489.96 ± 2.68	7491/86	402
Average Rank	1.8	**1.2**	**1.0**	2.0	-	-

Table 2: Comparison of AUC and training times for GGNE ($\alpha = 3\%$) and ChipClass (CC) across larger datasets.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

5 Conclusions

This work introduces the GGNE, a case study on ensemble learning techniques aimed at decreasing training time. It employs a graph-based SVM known as ChipClass. GGNE partitions data into smaller segments, constructs graphs individually, and aggregates predictions using majority voting. Experiments conducted on large datasets have proven that GGNE can make training up to 1500 times faster while maintaining competitive AUC performance. This indicates that significant efficiency gains can be made without sacrificing predictive quality. Future work will focus on adapting our ensemble approach to other classifiers with scalability challenges.

References

[1] Nathan C. Frey, Baolin Li, Joseph McDonald, Dan Zhao, Michael Jones, David Bestor, Devesh Tiwari, Vijay Gadepally, and Siddharth Samsi. Benchmarking resource usage for efficient distributed deep learning. In *2022 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–8, 2022.

[2] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[3] Daniel Asante Otchere, Tarek Omar Arbi Ganat, Raoof Gholami, and Syahrir Ridha. Application of supervised machine learning paradigms in the prediction of petroleum reservoir properties: Comparative analysis of ann and svm models. *Journal of Petroleum Science and Engineering*, 200, 2021.

[4] L.C.B. Torres, C.L. Castro, F. Coelho, F. Sill Torres, and A.P. Braga. Distance-based large margin classifier suitable for integrated circuit implementation. *Electronics Letters*, 51(24):1967–1969, November 2015. Publisher: Institution of Engineering and Technology (IET).

[5] K. Ruben Gabriel and Robert R. Sokal. A New Statistical Approach to Geographic Variation Analysis. *Systematic Zoology*, 18(3):259, September 1969.

[6] Piotr Jurkiewicz. TopoHub: A repository of reference Gabriel graph and real-world topologies for networking research. *SoftwareX*, 24:101540, December 2023.

[7] Asako Terasawa and Yoshihiro Gohda. Hidden order in amorphous structures: Extraction of nearest neighbor networks of amorphous Nd–Fe alloys with Gabriel graph analyses. *The Journal of Chemical Physics*, 149(15):154502, October 2018.

[8] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, October 2001.

[9] Leo Breiman. Pasting Small Votes for Classification in Large Databases and On-Line. *Machine Learning*, 36(1-2):85–103, July 1999. Publisher: Springer Science and Business Media LLC.

[10] Wan Zhang and Irwin King. A study of the relationship between support vector machine and Gabriel graph. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, pages 239–244, Honolulu, HI, USA. IEEE.