Hyperdimensional Classification

Margarita Geleta[1]

[1]University of California, Irvine, Department of Computer Science, Irvine, CA, USA

1 Part 1

To evaluate the Hyperdimensional Computing techniques for classification (HDC), in part 1, we are going to use the ISOLET (Isolated Letter Speech Recognition) dataset, which was generated as follows: 150 subjects spoke the name of each letter of the alphabet twice. Thus, there are 52 training examples from each speaker. The speakers have been grouped into sets of 30 speakers each, 4 groups can serve as training set, the last group as the test set. A total of 3 examples are missing, the authors dropped them due to difficulties in recording. Overall, the dataset results in 7797 samples and 617 features (Figure 1).

All features are continuous, real-valued attributes scaled into the range -1.0 to 1.0. The features include spectral coefficients; contour features, sonorant features, pre-sonorant features, and post-sonorant features[1]. The exact order of appearance of the features is not known. The classification task is to identify which letter did the speakers pronounce given the aforementioned features.

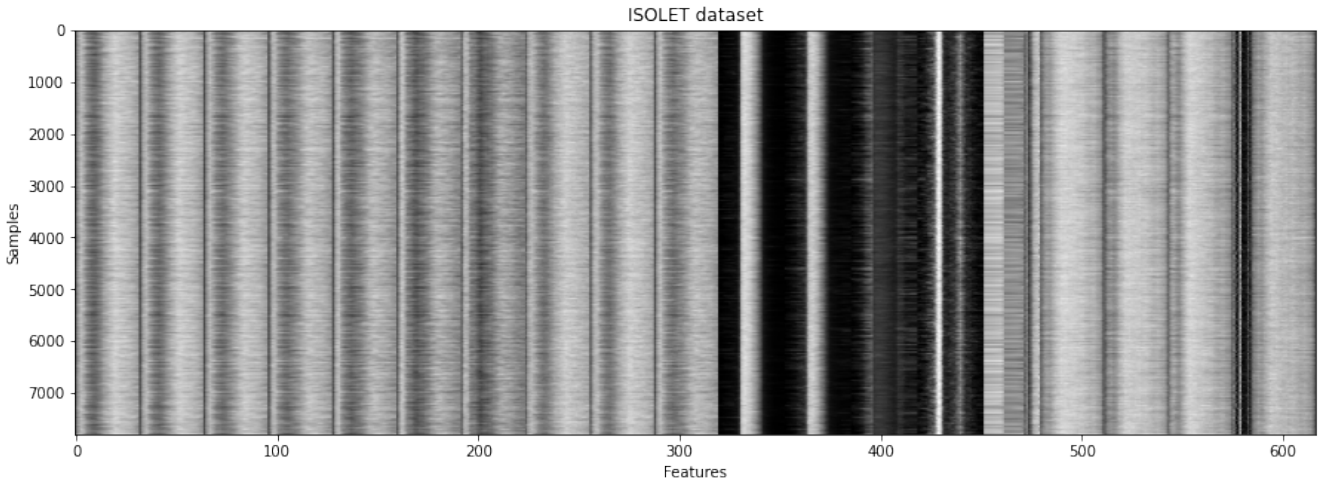

Figure 1. The ISOLET dataset.

1.1 Feature Vector Encoding

First of all, we need to encode the dataset. This encoding maps the data points from the input space to the high-dimensional space[2]. There are several encoding approaches, in this report we show the **permutation-based**, **association-based** and **kernel-based** encodings. Suppose we want to map a data point $\boldsymbol{x} \in \mathbb{R}^d$ to the hyperdimensional space, to obtain the hypervector $\boldsymbol{h} \in \{-1, +1\}^D$. We first find the maximum and minimum feature values and we quantize that range into m levels, obtaining a level set $\mathcal{L} = \{L_1, ..., L_m\}$. Let us define the hyperdimensional operators: bundling $\oplus$, binding $\otimes$, and permutation $\rho(\cdot)$.

1. **Permutation-based encoding**: also known as N-gram based encoder[3]. We assign a random hypervector to each level L_k. For each feature value in $\boldsymbol{x}$, we retrieve the corresponding level hypervector. For each sample $\boldsymbol{x}_i$, we obtain $\boldsymbol{h}_i$ by:

$$\boldsymbol{h}_i = h_1^i \oplus \rho h_2^i \oplus ... \oplus \rho^{d-1} h_d^i = \sum_{j=1}^{d} \rho^{d-1} h_j^i \tag{1}$$

Where $h_j^i \in \mathcal{L}$ is the hypervector corresponding to the value of the j-th feature of the i-th sample.

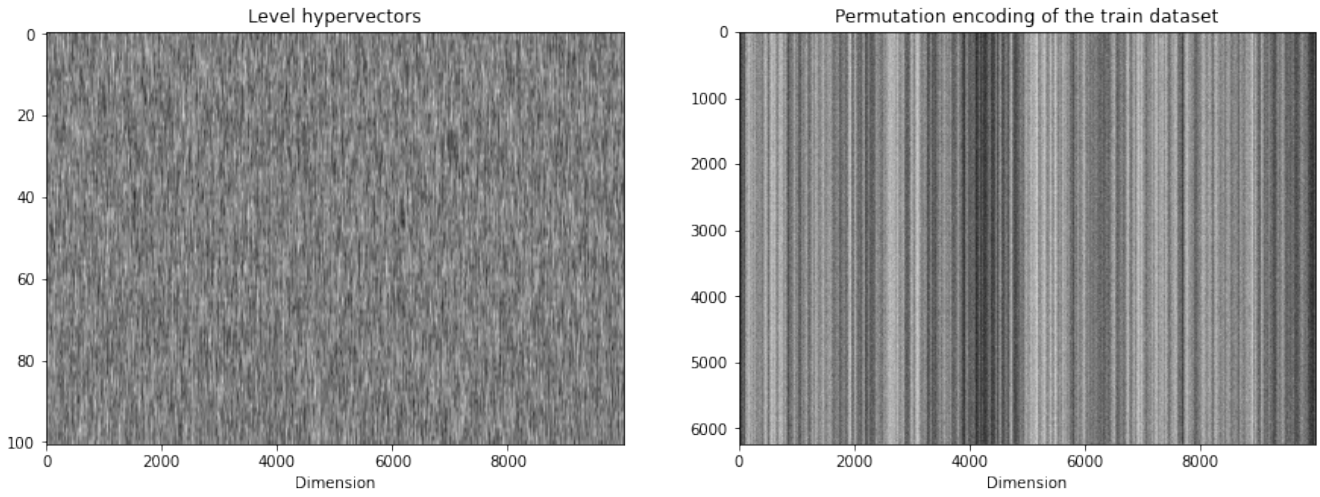

Figure 2. Permutation encoding with $D = 10,000$ of the ISOLET dataset.

2. **Association-based encoding**: also known as record-based encoder[3]. We assign a random hypervector ID to each feature and a random hypervector to each level L_k. For each feature value in $\boldsymbol{x}$, we retrieve the corresponding level hypervector. For each sample $\boldsymbol{x}_i$, we obtain $\boldsymbol{h}_i$ by:

$$\boldsymbol{h}_i = (h_1^i \otimes \mathrm{ID}_1) \oplus (h_2^i \otimes \mathrm{ID}_2) \oplus ... \oplus (h_d^i \otimes \mathrm{ID}_d) = \sum_{j=1}^{d} h_j^i \otimes \mathrm{ID}_j \tag{2}$$

Where $h_j^i \in \mathcal{L}$ is the hypervector corresponding to the value of the j-th feature of the i-th sample and ID_j is the hypervector identifier of the j-th feature.

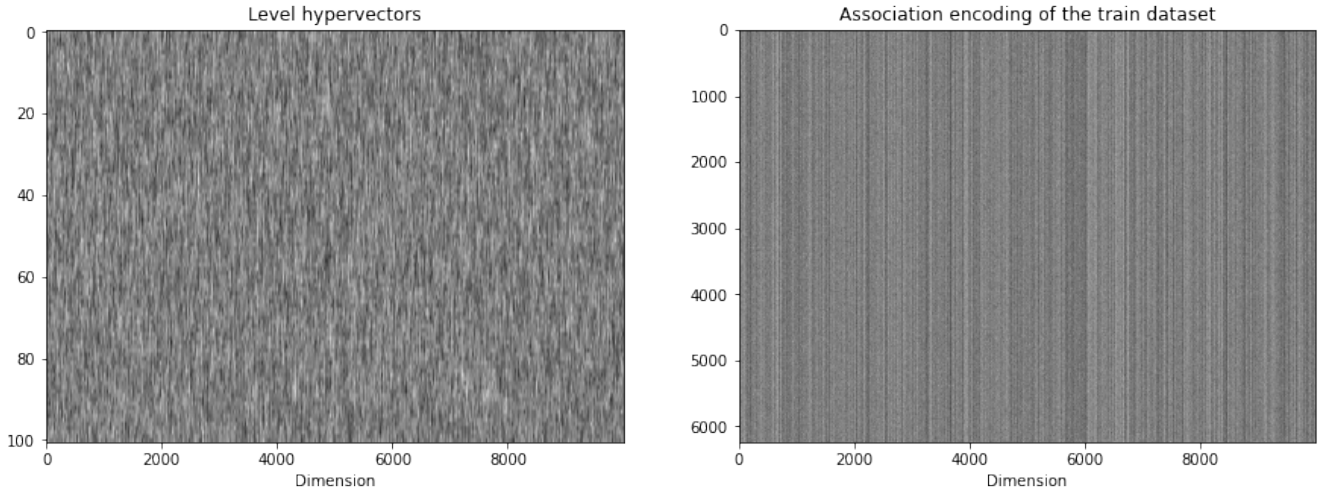

Figure 3. Association encoding with $D = 10,000$ of the ISOLET dataset.

3. **Kernel-based encoding**: in this report we have been using the hyperbolic tangent transfer function for the kernel method, although there exist many other alternatives (as the approximation of the RBF kernel). In the kernel method we define D basis hypervectors $\boldsymbol{B}_k$ sampled from the Gaussian distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$ for each dimension. Then, we bind each data point $\boldsymbol{x}_i$ with each basis hypervector. Adding the components of each resulting hypervector from the binding operation we obtain the k-th component of the D-dimensional encoding of $\boldsymbol{x}_i$. Finally, we apply the hyperbolic tangent operation on top of each component of the vector, this way obtaining the hypervector $\boldsymbol{h}_i$.

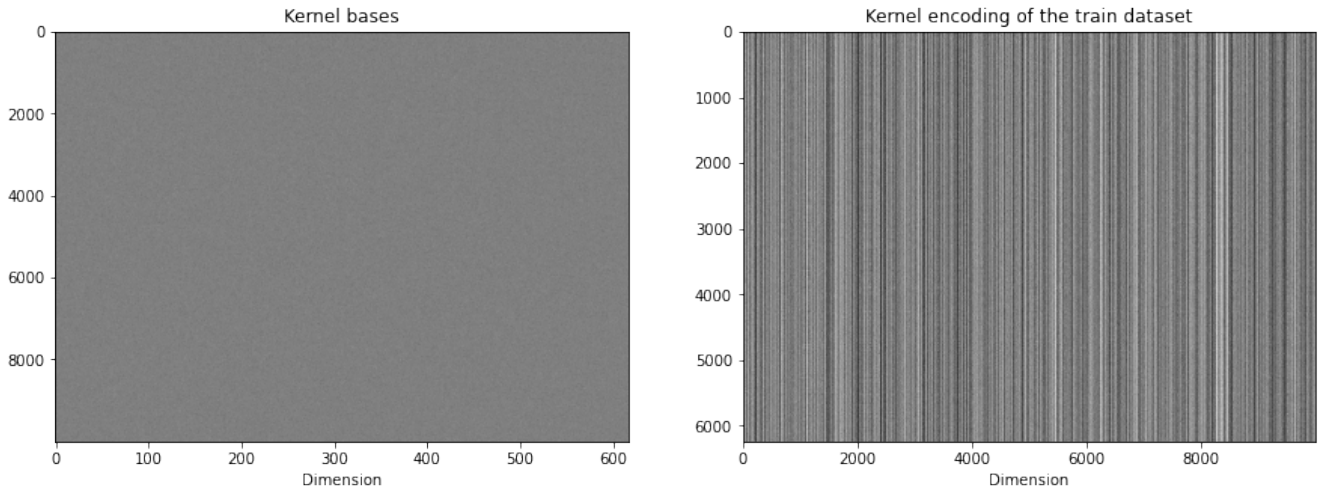

Figure 4. Kernel encoding with $D = 10,000$ of the ISOLET dataset using the hyperbolic tangent transfer function.

1.2 Training results

We trained several HDC models with different encodings (permutation-, association-, and kernel-based) and different training methods (single-pass, adaptive single-pass with learning rate $\alpha = 0.1$ and retraining with 20 iterations and batch size 1). In Figure 5 we show the 26 class hypervectors for the ISOLET dataset using the permutation encoding. In Figure 6 we show the same class hypervectors, after a retraining with 50 iterations and batch size 1. As it can be observed comparing Figures 5 and 6, retraining makes the hypervectors more holographic. In Figure 7 we show the pairwise Manhattan distance between class vectors using two different encodings (permutation- and kernel-based). It can be observed that the encodings preserve the natural differences found in spelling between the letters.

In Table 1, we report the classification accuracy of different encoding and models on ISOLET test data.

		Encoding		
		Permutation	Association	Kernel
Training Method	Single-Pass	69.55%	72.82%	86.15%
	Adaptive Single-Pass with lr=0.1	79.10%	78.20%	86.67%
	Retraining with 20 iterations and batch size 1	81.28%	82.62%	93.20%

Table 1. Test accuracy using different encodings and training methods.

1.3 Retraining method

In Figure 8 we report the test accuracy on the ISOLET dataset vs. the number of iterations of the retraining method. After 10 iterations, the accuracy gets stable. At the end of 20 iterations, we end up with 81.28%, 82.62% and 93.20% test accuracies for permutation-, association- and kernel-based encodings.

1.4 Batch size effect

Batch size in the context of hyperdimensional computing training refers to the delay with which we update the class hypervectors. In our experiments, using the retraining method, using batch sizes $\{1, 5, 10, 15, 25, 50, 100, 500\}$ converged to the same test accuracy within each corresponding encoding, i.e., to 85.19%, 86.79% and 92.37% using permutation-, association- and kernel-based encodings; from what we can conclude that the batch size has no significant impact on the accuracy.

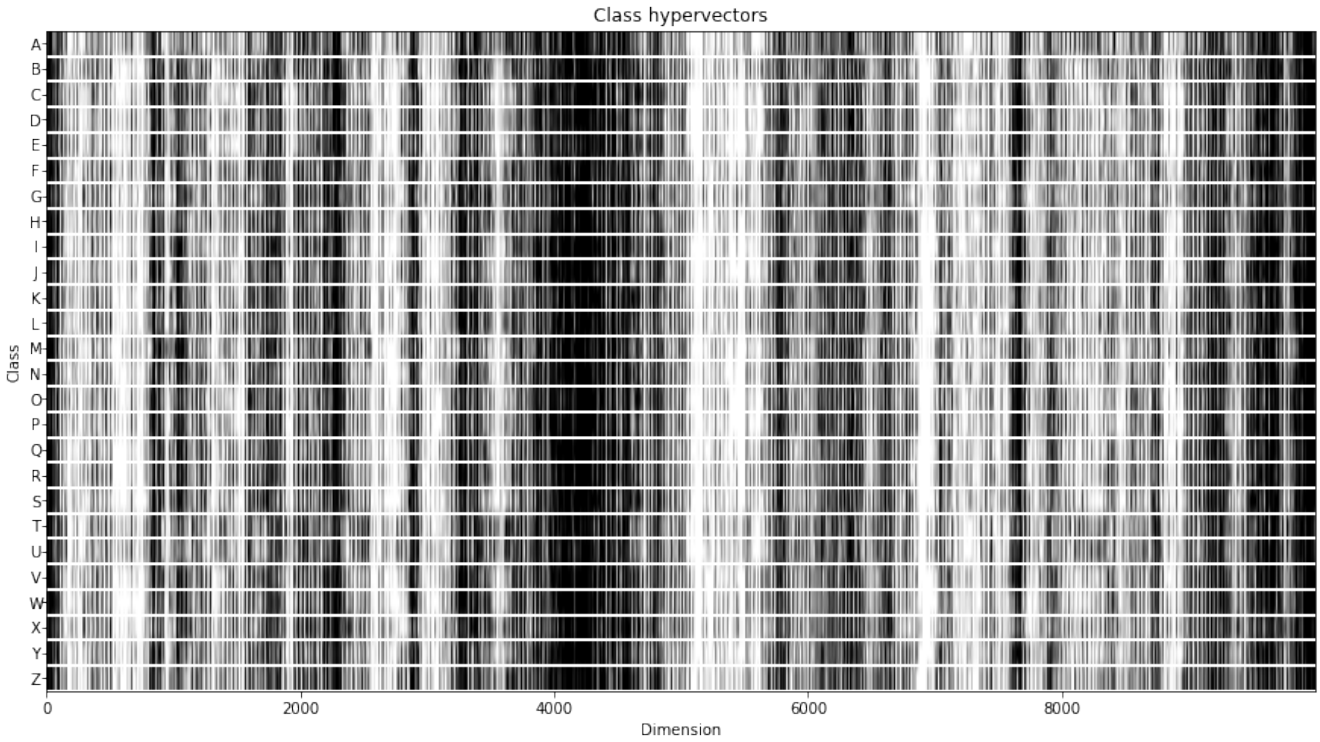

Figure 5. Class hypervectors using the permutation encoding with $D = 10,000$ dimensions.

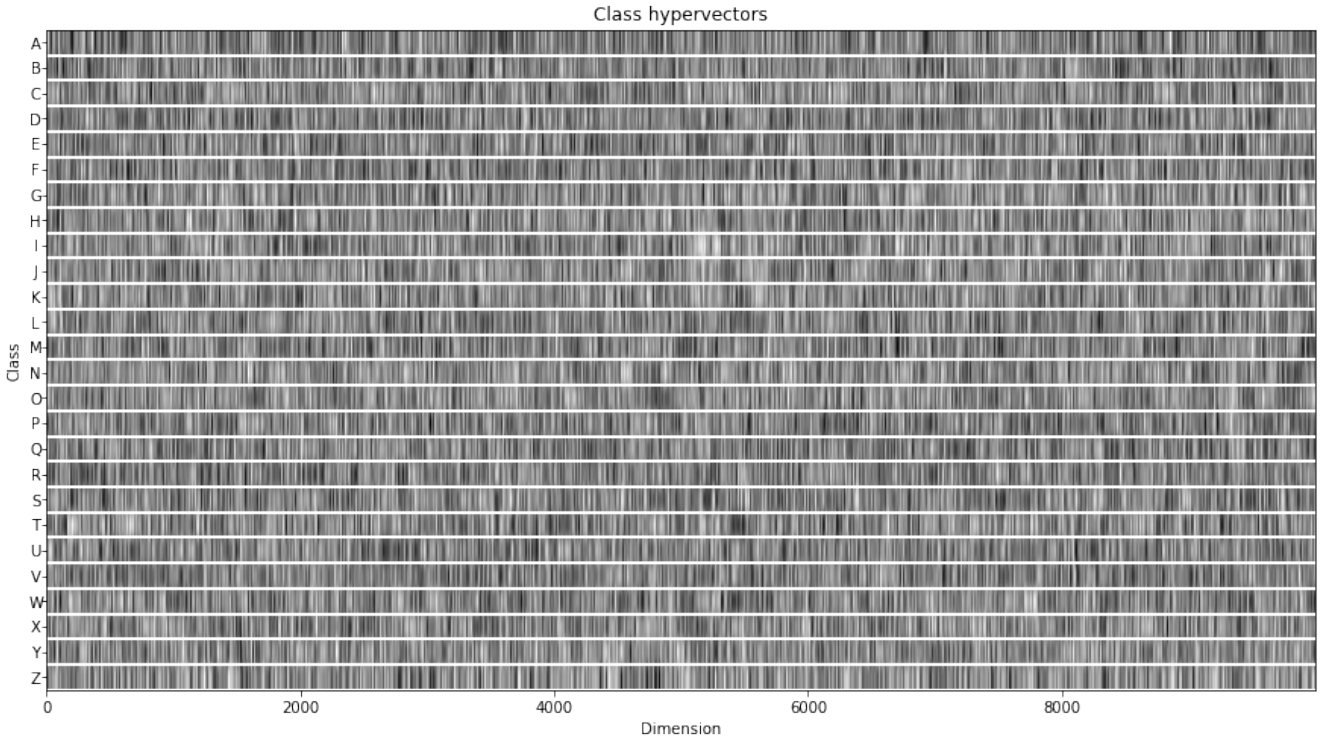

Figure 6. Class hypervectors using the permutation encoding with $D = 10,000$ dimensions retrained in 50 iterations.

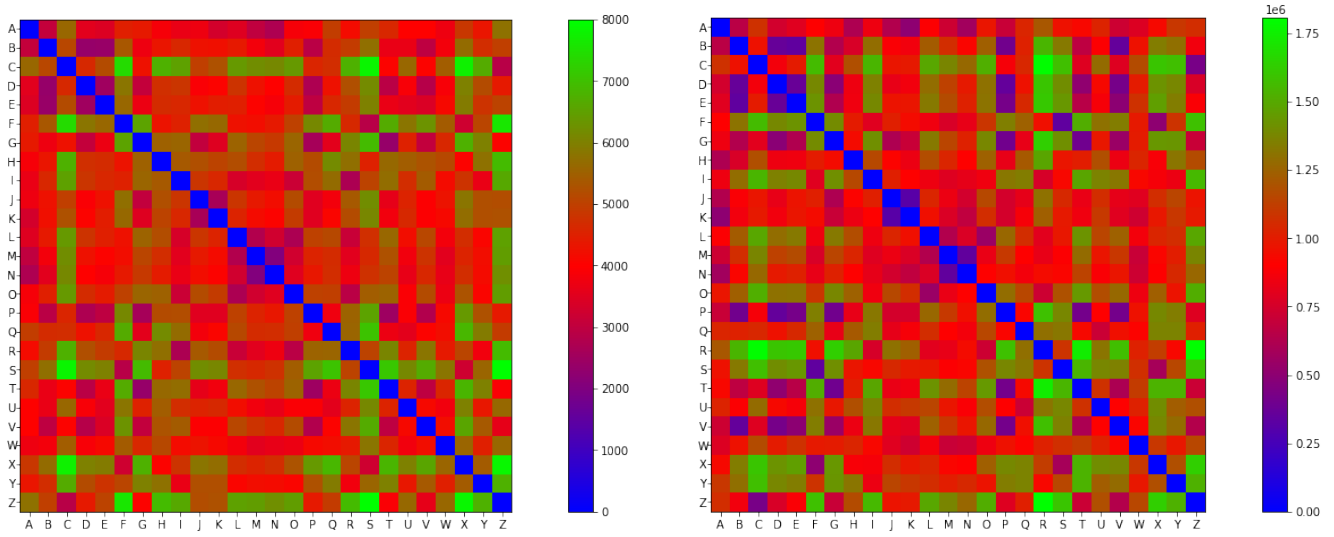

Figure 7. Manhattan distance computed between class hypervectors using the permutation-based (at the left) and kernel-based (at the right) encodings. Close to blue hues corresponds to high similarity (low dissimilarity) and green hues point low similarity (high dissimilarity). Hypervectors are capable of capturing the similarity between the spelling of the letters. For instance, "E" is similarly pronounced as "C", "D" and "P", but is quite different to "F" and "S". Both "F" and "S" are similar pairwises.

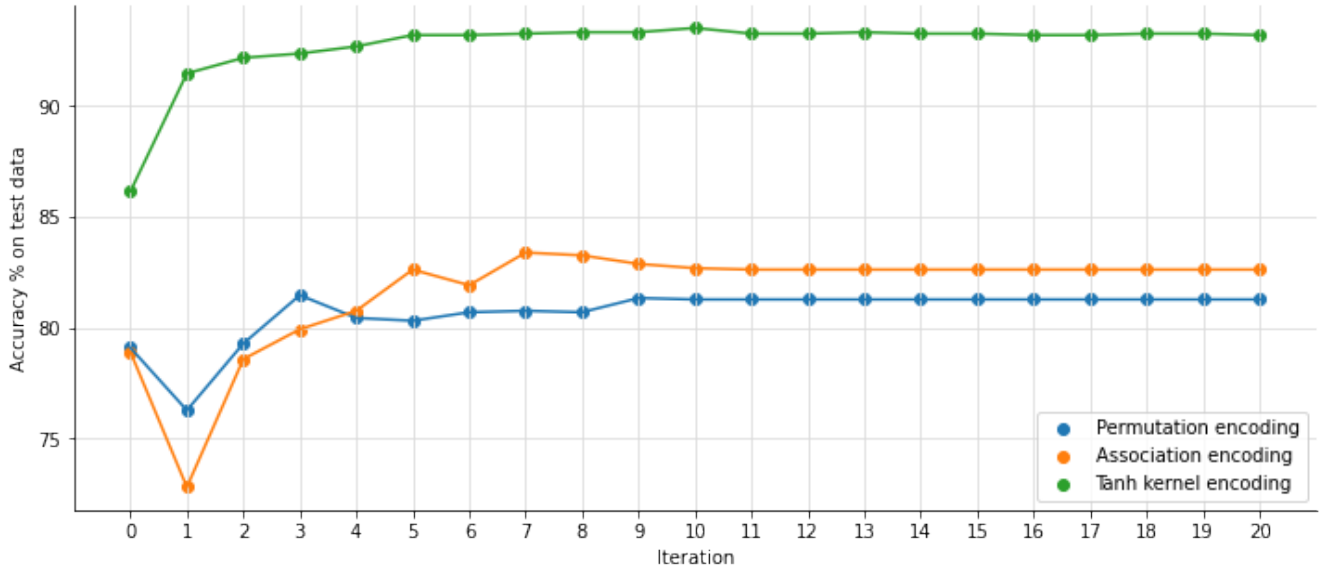

Figure 8. The progression of accuracy w.r.t. to the number of iterations using the retraining method for each of the 3 encodings used in this report.

1.5 Dimensionality of the hypervectors

In Figure 9 and Table 2 we report HDC classification accuracy using different dimensionality for the hypervectors for all encoding using the retraining method. The results do not show significant differences or explicit trends.

2 Part 2

The data from EEG dataset comes from a large study to examine how EEG correlates with the genetic predisposition to alcoholism. It contains measurements from 64 electrodes placed on subject's scalps which were sampled at 256 Hz (3.9-msec epoch) for 1 second. There were two groups of subjects: alcoholic and control. Each subject was exposed

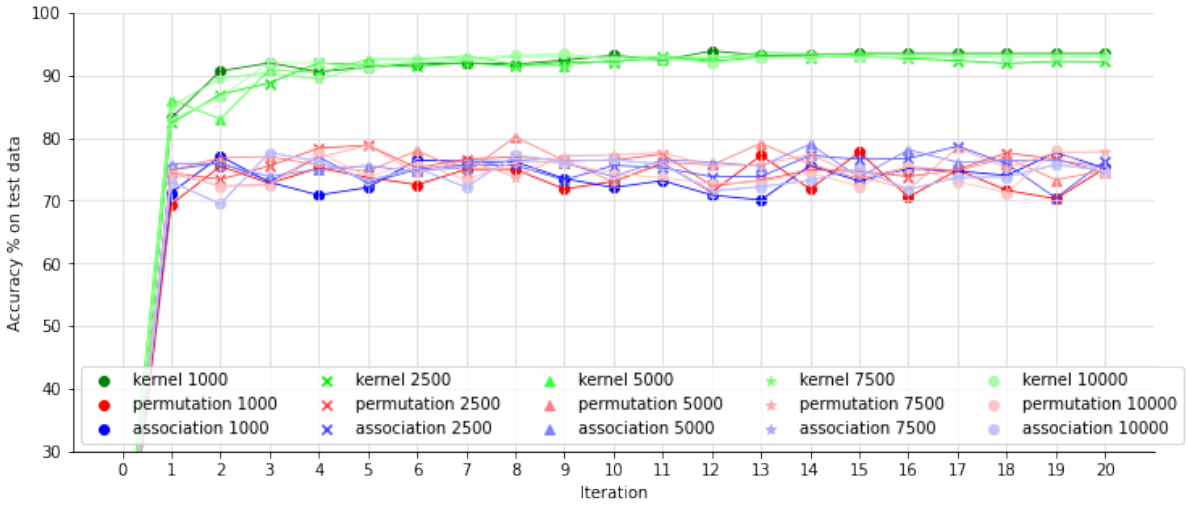

Figure 9. Class hypervectors using the permutation encoding with $D = 10,000$ dimensions retrained in 50 iterations.

		Encoding		
		Permutation	Association	Kernel
Dimensionality	1000	75.19%	75.19%	93.52%
	2500	74.42%	76.34%	92.17%
	5000	74.87%	75.32%	93.46%
	7500	77.82%	74.80%	93.01%
	10000	74.42%	74.80%	92.94%

Table 2. Effect of the hypervectors' dimensionality on the test accuracy using different encodings.

to either a single stimulus (S1) or to two stimuli (S1 and S2) which were pictures of objects chosen from the 1980 Snodgrass and Vanderwart picture set. When two stimuli were shown, they were presented in either a matched condition where S1 was identical to S2 or in a non-matched condition where S1 differed from S2.

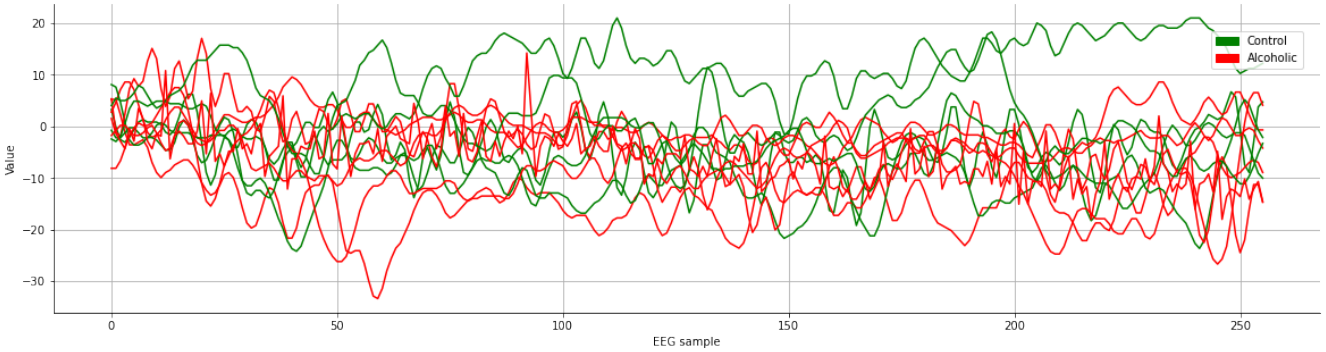

Figure 10. The EEG dataset.

Because each individual has 256 EEG samples, we exclusively use window sizes being powers of 2, so that the window size is divisible bt 256. We report the classification accuracy using different sampling windows sizes and dimensionality in Table 3. All encodings are permutation-based, using 100 levels, with maximum value being 50 and the minimum one -50. This values have been chosen by looking at the distribution of the feature values, which can be observed in the histogram of Figure 12. Observing the results from Table 3 we can infer that extremely small and large window sizes do not help in improving the accuracy, there is sort of trade-off between the size of the window and the information captured. From the experiments, we can see that the best window size is 8.

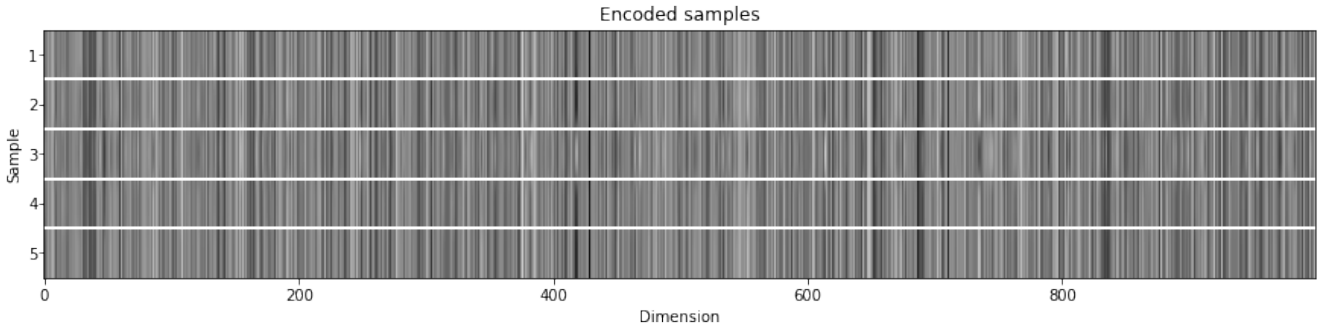

Figure 11. Several samples from the EEG dataset encoded with a permutation-based encoder with dimensionality 1000, 100 levels, flip rate 25, and window size 64.

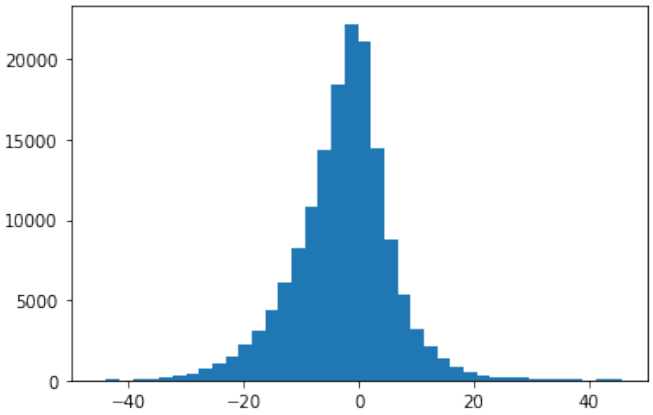

Figure 12. Histogram of the feature values from the EEG dataset.

		Window size				
		2	4	8	16	64
Dimensionality	1000	64.34	68.56	73.29	72.18	67.05
	2500	66.38	69.43	75.46	74.56	72.67
	5000	67.45	70.02	75.32	73.88	72.68
	7500	66.89	70.42	76.28	75.90	74.78
	10000	67.22	69.89	74.98	76.81	73.04

Table 3. Classification accuracy on EEG test data using different window sizes and dimensionality.

References

1. R. A. Cole and M. A. Fanty, "Spoken letter recognition," in *HLT*, 1990.
2. A. Hernandez-Cane, N. Matsumoto, E. Ping, and M. Imani, "Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system," in *2021 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 56–61, 2021.
3. M. Imani, C. Huang, D. Kong, and T. Rosing, "Hierarchical hyperdimensional computing for energy efficient classification," in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2018.