

Nearest Neighbor CCP-Based Molecular Sequence Analysis

An Efficient Approach for High-Dimensional Biological Data

Sarwan Ali¹, Prakash Chourasia², Bipin Koirala³, Murray Patterson²

IEEE Transactions on Computational Biology and Bioinformatics (TCBB) 2025

¹Columbia University ²Georgia State University ³Georgia Institute of Technology

Outline

Problem Formulation & Motivation

Correlated Clustering and Projection (CCP)

Proposed CCP-NN Algorithm

Complexity Analysis

Convergence Analysis

Experimental Validation

Technical Insights & Future Directions

Problem Formulation & Motivation

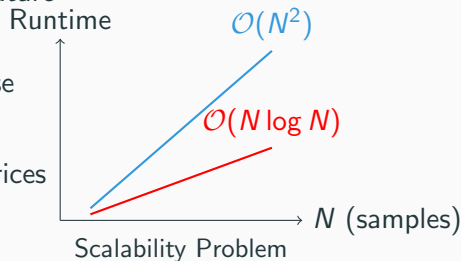
Computational Challenges in Molecular Sequence Analysis

High-Dimensional Data Challenges:

- **Curse of Dimensionality:** Feature spaces $\mathbb{R}^M$ with $M \gg N$
- **Sparsity:** Biological sequences create sparse feature representations
- **Computational Complexity:** $\mathcal{O}(N^2M)$ pairwise computations
- **Memory Requirements:** $\mathcal{O}(N^2)$ distance matrices

Existing DR Limitations:

- Linear methods (PCA, LDA): Assume linear relationships
- Non-linear methods: Computationally prohibitive for large N
- CCP: Promising but $\mathcal{O}(N^2)$ bottleneck



Correlated Clustering and Projection (CCP)

CCP Algorithm: Mathematical Foundation

Input: Dataset $\mathbf{X} \in \mathbb{R}^{N \times M}$, where N = samples, M = features

Objective: Find low-dimensional embedding $\phi_{CCP} : \mathbb{R}^M \rightarrow \mathbb{R}^{n_c}$

CCP Pipeline

1. **Variance-based Feature Selection:** Select top f features where $f = \lfloor \text{numCutoff} \times |\{j : \text{Var}(X_j) > 0\}| \rfloor$
2. **K-means Clustering:** Partition features into $(n_c - 1)$ clusters
3. **Density Map Computation:** For cluster C_i with features $\mathcal{F}_i$:

$$\rho_i(\mathbf{x}) = \sum_{j,k \in \mathcal{F}_i} K \left(\frac{d(\mathbf{x}_j, \mathbf{x}_k)}{\text{scale}} \right) \quad (1)$$

where $K(z) = e^{-(z)^p}$ (Exponential) or $K(z) = \frac{1}{1+z^p}$ (Lorentz)

CCP Density Computation: Algorithmic Details

Algorithm 1 CCP Correlation Computation

```
1: function COMPUTECORRELATIONS(idx_comp, idx_Feat, X, transform)
2:   if transform then
3:     D  $\leftarrow$  pairwise_distances(X[:, idx_Feat], Xref[:, idx_Feat])
4:   else
5:     D  $\leftarrow$  pairwise_distances(Xref[:, idx_Feat])
6:   end if
7:   if avgmindist[idx_comp] = 0 then
8:     avgmindist[idx_comp]  $\leftarrow \frac{1}{N} \sum_{i=1}^N \min_{j \neq i} \mathbf{D}_{ij}$ 
9:   end if
10:  if cutoff[idx_comp] = 0 then
11:     $\mu_D \leftarrow \text{mean}(\mathbf{D})$ ,  $\sigma_D \leftarrow \text{std}(\mathbf{D})$ 
12:    cutoff[idx_comp]  $\leftarrow \mu_D + 3\sigma_D$ 
13:  end if
14:  scale  $\leftarrow$  user_scale  $\times$  avgmindist[idx_comp]
15:   $\rho \leftarrow \text{computeDensity}(\mathbf{D}, \text{scale}, \text{cutoff}[\text{idx\_comp}])$ 
16:  return  $\rho$ 
17: end function
```

Proposed CCP-NN Algorithm

CCP-NN: Approximate Nearest Neighbor Enhancement

Key Innovation: Replace exact pairwise distances with ANN search

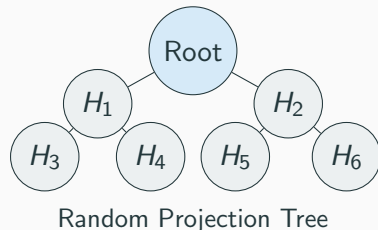
AnnoyIndex Properties:

- **Data Structure:** Forest of random projection trees
- **Build Complexity:** $\mathcal{O}(N \log N \cdot f)$
- **Query Complexity:** $\mathcal{O}(\log N)$ per query
- **Space Complexity:** $\mathcal{O}(N \cdot f)$

Approximation Quality:

$$\mathbb{E}[||\mathcal{N}_k(x_i) - \tilde{\mathcal{N}}_k(x_i)||] \leq \epsilon \quad (2)$$

Modification: Only Step 5 (density computation) changes; Steps 1-4 and 6-7 remain identical



CCP-NN Algorithm: Technical Implementation

Algorithm 2 CCP-NN Nearest Neighbor Correlation Computation

```
1: function NEARESTNEIGHBORCOMPUTECORR(idx_comp, idx_Feat, X, transform)
2:   idx  $\leftarrow$  AnnoyIndex(len(idx_Feat), metric)
3:   for i = 0 to len(Xref[:, idx_Feat]) do
4:     idx.add_item(i, Xref[:, idx_Feat][i])
5:   end for
6:   idx.build(-1) ▷ Build index with all trees
7:   if transform then
8:      $\rho \leftarrow [\text{idx.get\_nns\_by\_vector}(\text{feat}, k) \text{ for } \text{feat} \text{ in } \mathbf{X}[:, \text{idx\_Feat}]]$ 
9:   else
10:     $\rho \leftarrow [\text{idx.get\_nns\_by\_item}(i, k) \text{ for } i \text{ in range}(N)]$ 
11:   end if
12:    $\rho \leftarrow \text{reshape}(\rho, (-1, 1))$  ▷ Same scaling and cutoff computation as CCP
13:   scale  $\leftarrow$  user_scale  $\times$  avgmindist[idx_comp]
14:    $\rho \leftarrow \text{computeDensity}(\rho, \text{scale}, \text{cutoff}[\text{idx\_comp}])$ 
15:   return  $\rho$ 
16: end function
```

Complexity Analysis

Theoretical Complexity Comparison

Algorithm	Time Complexity	Space Complexity
CCP	$\mathcal{O}(N \cdot f(n_c \cdot n_{iter} + N))$ Dominated by $\mathcal{O}(N^2 \cdot f)$	$\mathcal{O}(N(M + N))$
CCP-NN	$\mathcal{O}(N \cdot f(n_c \cdot n_{iter} + \log N))$ Dominated by $\mathcal{O}(N \log N \cdot f)$	$\mathcal{O}(N(M + \log N \cdot f))$

Asymptotic Improvement:

$$\text{Speedup Factor} = \frac{\mathcal{O}(N^2)}{\mathcal{O}(N \log N)} = \frac{N}{\log N} \quad (3)$$

Concrete Example: For $N = 10^4$, $f = 100$, $n_c = 10$:

- CCP: $\sim 10^9$ operations
- CCP-NN: $\sim 1.3 \times 10^6$ operations
- **Theoretical speedup:** $\sim 770\times$

Convergence Analysis

Theoretical Convergence Guarantees

Setup: Let $\mathbf{X} \sim \mathbb{P}(\mathbf{X})$ with density $p(\mathbf{x})$

Approximation Error: AnnoyIndex provides ϵ -approximate nearest neighbors:

$$\mathbb{E}[||\mathcal{N}_k(\mathbf{x}_i) - \tilde{\mathcal{N}}_k(\mathbf{x}_i)||] \leq \epsilon \quad (4)$$

Density Estimation: CCP-NN density estimate:

$$\hat{p}(\mathbf{x}_i) = \frac{1}{hk} \sum_{\mathbf{x}_j \in \tilde{\mathcal{N}}_k(\mathbf{x}_i)} K\left(\frac{\mathbf{x}_i - \mathbf{x}_j}{h}\right) \quad (5)$$

Error Bound: Using triangle inequality:

$$|\hat{p}(\mathbf{x}_i) - p(\mathbf{x}_i)| \leq |\hat{p}(\mathbf{x}_i) - p(\mathbf{x}_i, \tilde{\mathcal{N}}_k(\mathbf{x}_i))| + |p(\mathbf{x}_i, \tilde{\mathcal{N}}_k(\mathbf{x}_i)) - p(\mathbf{x}_i)| \quad (6)$$

Final Result:

$$\mathbb{E}[|\hat{p}(\mathbf{x}_i) - p(\mathbf{x}_i)|] \leq \mathcal{O}\left(\epsilon + h^4 + \frac{1}{kh}\right) \quad (7)$$

Experimental Validation

Datasets and Experimental Setup

Dataset	Samples	Classes	Type	Avg. Length
Protein Subcellular	5,959	11	Protein	934
Coronavirus Host	5,558	21	Spike Protein	1,273
Human DNA	4,380	7	Nucleotide	4,380

Embedding Methods:

- **OHE**: One-Hot Encoding
- **Spike2Vec**: Word2Vec-based protein embeddings
- **PWM2Vec**: Position Weight Matrix embeddings
- **Autoencoder**: Neural network-based embeddings

Evaluation: 7 classifiers $\times$ 5-fold CV $\times$ 5 random runs = 175 experiments per method

Performance Results: Accuracy & Runtime

Method	Accuracy	Δ
OHE + CCP	0.712	-
OHE + CCP-NN	0.752	+5.6%
Spike2Vec + CCP	0.784	-
Spike2Vec + CCP-NN	0.812	+3.6%
PWM2Vec + CCP	0.801	-
PWM2Vec + CCP-NN	0.834	+4.1%
Autoencoder + CCP	0.823	-
Autoencoder + CCP-NN	0.859	+4.4%

Table 1: Protein Subcellular Dataset

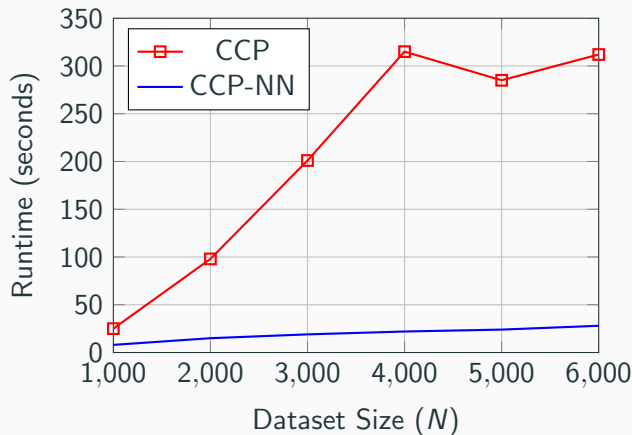
Method	Runtime (s)	Speedup
OHE + CCP	124.5	-
OHE + CCP-NN	95.5	1.3×
Spike2Vec + CCP	87.2	-
Spike2Vec + CCP-NN	24.1	3.6×
PWM2Vec + CCP	203.7	-
PWM2Vec + CCP-NN	89.8	2.3×
Autoencoder + CCP	315.4	-
Autoencoder + CCP-NN	22.5	14×

Table 2: Runtime Comparison

Key Findings:

- Consistent accuracy improvements across all embedding methods
- Significant runtime reductions, especially with dense embeddings
- Better approximation quality leads to improved classification performance

Scalability Analysis



Scaling Behavior:

CCP shows quadratic growth pattern, while CCP-NN exhibits near-linear scaling.

Memory Usage:

- CCP: $\mathcal{O}(N^2)$ distance matrices
- CCP-NN: $\mathcal{O}(N \log N)$ index structure

Practical Impact: For $N > 10^4$, CCP becomes computationally prohibitive, while CCP-NN remains tractable.

Technical Insights & Future Directions

Key Technical Contributions

Algorithmic Innovation

- **Hybrid:** CCP's correlation-based clustering while leveraging ANN efficiency
- **Theoretical Guarantees:** Formal converg. analysis with bounded approx. error
- **Practical Scalability:** Reduces complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$

Empirical Validation

- **Accuracy Improvement:** Up to 10.8% better classification performance
- **Runtime Efficiency:** Up to 14 $\times$ speedup on real biological datasets
- **Robustness:** Consistent improvements across multiple embedding methods

Future Research Directions

- **Deep Integration:** Incorporation with transformer-based protein language models
- **Multi-scale Analysis:** Extension to hierarchical biological sequence structures
- **Distributed Computing:** Parallel implementation for genome-scale datasets

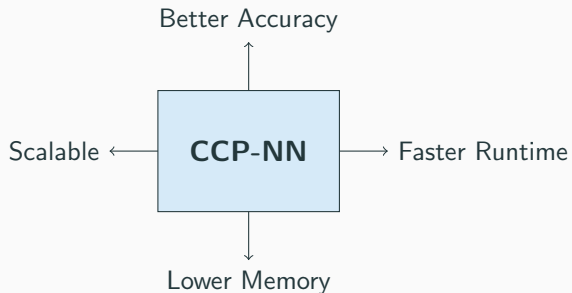
CCP-NN: Bridging Theory and Practice

Theoretical Contributions:

- Rigorous complexity analysis
- Convergence guarantees
- Approximation error bounds

Practical Impact:

- Enables large-scale sequence analysis
- Maintains classification quality
- Significant computational savings



Thank You!

Questions & Discussion

`sa4559@cumc.columbia.edu`