Low-Complexity Compressed Analysis in Eigenspace with Limited Labeled Data for Real-Time Electrocardiography Telemonitoring

Kai-Chieh Hsu, Bo-Hong Cho, Ching-Yao Chou and An-Yeu (Andy) Wu

National Taiwan University
ECG Telemonitoring with Edge Computing

- Mobile Telemedicine with Wireless Body Area Network (WBAN) [1]
  - Patient-centered health-care
  - Ubiquitous health-care

- ECG Telemonitoring [2], [3]
  - Record the electrical activity of the heart
  - Standard practice in hospitals for diagnoses

- Edge Computing [4]
  - bandwidth cost saving
  - battery life constraint
  - latency requirement
Edge Computing under Existing IoT Systems

Wireless Body Area Network

Massive Sampled Data

Transmit

Existing Module

Receive

Machine Learning

Massive Sampled Data

Edge-Devices
Compressed Sensing for ECG Telemonitoring

- Problems of Digital Wavelet Transform (DWT)
  - High bandwidth incompatible to ADC (Nyquist sample rate)
  - High Computational Complexity (Compression)

- Compressed sensing (CS) combines sampling and compressing
  - Reduce cost and latency in sampling
  - CS-based sensors achieves a 37% node lifetime extension [2]
Compressed Analysis for ECG Telemonitoring

- **Reconstructed Analysis (RA)**
  - High computational complexity because of CS reconstruction algorithms
  - Inappropriate at edge devices.

- **Compressed Analysis (CA)**
  - Reduce power (classification on compressed signals), suitable at edge devices
  - Reduce the bandwidth requirement (only transmitting AF signals)

AF: Atrial Fibrillation
Naïve CA (CA-N)

- Combining CS with Task-Driven Dictionary Learning (TDDL)
  - What is TDDL [5]
    - Learning a dictionary (D) to provide predictive sparse coding (\(\alpha\)) at given data set
    - Learning a classifier (W) to classify by the sparse coding \(\alpha\)
  - Why we choose TDDL?
    - Low Complexity → Overcome battery constraint and bandwidth scarcity
    - High Generalization → Limited label of ECG dataset

- The on-line inference mode of CA-N
  - D and W learned on original data (X)
  - Accuracy degrades, needing double parameters to reach same performance on original data
Contribution of Proposed Scheme (1/2)

- Low-Complexity (overcame battery and bandwidth requirement)
  - Our proposed Eigenspace-aided Compressed Analysis (CA-E) vs Naïve Compressed Analysis (CA-N)

<table>
<thead>
<tr>
<th>Model</th>
<th># Parameters</th>
<th>Training Time (s)</th>
<th>Inference Time (ms)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-N</td>
<td>13k</td>
<td>452.56</td>
<td>26.94</td>
<td>89.24 ± 0.520</td>
</tr>
<tr>
<td>CA-E (Our proposed)</td>
<td>4.25k</td>
<td>107.15</td>
<td>3.50</td>
<td>90.05 ± 0.256</td>
</tr>
</tbody>
</table>

- Reduce about 67% parameters (Memory ↓)
- Reduce about 87% inference time (Power ↓)
- Reduce about 76% training time (Power ↓)
Contribution of Proposed Scheme (2/2)

- High-Stability
  - CA-E outperforms DNN and SVM by over 10% when the amount of data is halved. (Overcame limited label of ECG dataset)
  - CA-E reaches about 90% under all compressed ratio (Stable under all compressed ratio)
Eigenspace-Aided CA (Training)

- Principal Component Analysis (PCA)
  - Record mean vector (μ) of dataset (X)
  - Learn eigenspace (Ψ ∈ ℝ^{N×r}) of X
  - Transpose to eigenspace by T = Ψ^T(X - μ)

- TDDDL to learn D and W on T
  - Stage I. Initialize
    - Dictionary: online dictionary learning (ODL) [6]
    - Weight: square / logistic loss
  - Stage II. Co-optimize D and W with labels
    - Alternates between A and D, W
    - Update dictionary with back propagation rule
  - Sparse coding plays an important role in both stage.

\[
\alpha_D \triangleq \arg\min_{\alpha \in \mathbb{R}^d} \frac{1}{2} \|x - D\alpha\|_2^2 + \lambda \|\alpha\|_1
\]
Eigenspace-Aided CA (Inference)

- **Eigenspace Transform**
  - Compressed sensing signal is transmitted with known sensing matrix (\( \Phi \)), the decoding data is obtained by
  \[
  s = (\Phi \Psi)^+ (\hat{x} - \Phi \mu) = \Theta^+ (\hat{x} - \Phi \mu)
  \]
  - \((\quad)^+\): pseudo-inverse

- The decoding vector \((s)\) then pass through TDDL-based classifier
  - Get sparse coding \(\alpha(s, D)\)
  - Simple linear classifier \(W\)

\[
\alpha_D \triangleq \arg\min_{\alpha \in \mathbb{R}^d} \frac{1}{2} \| x - D \alpha \|_2^2 + \lambda \| \alpha \|_1
\]
Scheme Development (1/2)

- Sensor
  - DWT-Based
  - CS-Based

- Analysis
  - Reconstructed (RA)
  - Compressed (CA)
Scheme Development (2/2)

- Compressed Analysis
  - Prototype
    -バルス
    - Prototype
    - Compressed Analysis
    - Normal
    - AF
    - Reconstruction
    - Cloud
  - Naïve
    - 
    - Normal
    - AF
    - Accuracy Degrade
  - Eigenspace-aided
    - 
    - Normal
    - AF
    - Eigenspace
    - Dictionary
    - Classifier
Simulation Results (1/3)
Different Dictionary Size

- **Accuracy vs Dictionary Size**
  - To surpass DNN & SVM (~85%), CA-E needs **30 atoms**, but CA-N needs 60 atoms.
  - Under same number of atoms, CA-E outperforms CA-N by about 7%.

- **CA-E-50 vs. CA-N-100**
  - Reduce about **67%** parameters (Memory ↓)
  - Reduce about **87%** inference time (Power ↓)
  - Reduce about **76%** training time (Power ↓)

### Simulation Results (1/3)

<table>
<thead>
<tr>
<th>Model</th>
<th># Parameters</th>
<th>Training Time (s)</th>
<th>Inference Time (ms)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-N (100)</td>
<td>13k</td>
<td>452.56</td>
<td>26.94</td>
<td>89.24 ± 0.520</td>
</tr>
<tr>
<td>CA-E (50)</td>
<td>4.25k</td>
<td>107.15</td>
<td>3.50</td>
<td>90.05 ± 0.256</td>
</tr>
</tbody>
</table>
Simulation Results (2/3) Different Data Set Size

- CA-E is **more immune** to limited data challenge (ex. $N_r \leq 0.5$)
  - SVM and DNN dramatically drops below 80%
  - CA still maintain the performance
  - CA-E outperforms CA-N in **7% margin** when the number of atoms is the same.
Simulation Results (3/3)
Different Compressed Ratio

- CA-E can achieve about 90% accuracy under all compressed ratios
  - CA-N requires 100 atoms to achieve same level of performance
  - SVM and DNN have only about 80%
- CA-E is robust and address the entailed problems of variation of compress ratio
Conclusion

- We propose an eigenspace-aided compressed analysis for ECG telemonitoring, using
  - PCA to mitigate the influence of sensing matrix and reduce the dimension
  - TDDL to learn predictive sparse coding at eigenspace.

- The proposed eigenspace-aided compressed analysis achieves
  - Low complexity
  - High generalization
  - High stability of different compressed ratios
Reference


Thanks for your attention

Q&A
Backup
Experimental Setting

- ECG signals were recorded from the intensive care unit (ICU) of stroke in National Taiwan University Hospital (NTUH)
  - 231 normal records and 58 AF records (labeled by doctors)
  - Sample Frequency: 512 Hz
  - Each record randomly sample 2250 seconds
    - 1250 for training
    - 1000 for testing

- CS setting
  - Entries of sensing matrix: Bernoulli (0.5)

- Simulation Environment
  - Measured on Intel i5-4200M CPU @ 2.5 GHz
  - Using Python3

<table>
<thead>
<tr>
<th>TABLE I: Parameters setting for learning models</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-E and CA-N</td>
</tr>
<tr>
<td>$\ell_1$-Constraint ($\lambda_1$)</td>
</tr>
<tr>
<td>$[0.2, 0.5, 0.8]$</td>
</tr>
<tr>
<td>Regularization ($\nu$)</td>
</tr>
<tr>
<td>$[10^{-5}, 10^{-4}]$</td>
</tr>
<tr>
<td>SVM</td>
</tr>
<tr>
<td>Kernel</td>
</tr>
<tr>
<td>Radial Basis Function</td>
</tr>
<tr>
<td>$[0.08, 0.10, 0.12, 0.15, 0.2]$</td>
</tr>
<tr>
<td>Gamma ($\gamma$)</td>
</tr>
<tr>
<td>$[0.08, 0.10, 0.12, 0.15, 0.2]$</td>
</tr>
<tr>
<td>Cost ($C$)</td>
</tr>
<tr>
<td>$[500, 800, 1000]$</td>
</tr>
<tr>
<td>DNN</td>
</tr>
<tr>
<td>Hidden Layer Dimension</td>
</tr>
<tr>
<td>$[(16,32), (32,64), (64,128), (128,256), (8,16,32), $</td>
</tr>
<tr>
<td>$(16,32,64), (32,64,128)]$</td>
</tr>
</tbody>
</table>
Analysis of CA-N

- We need to increase the number of atoms in dictionary to compensate the performance degrade

- Figures below also present the sparse codings in original domain as comparison group.
## Comparison of CA-E and CA-N

- **CA-E-50 vs. CA-N-100**
  - Reduce about 67% parameters
  - Reduce about 76% training time,
  - Reduce about 87% inference time
  - Smaller performance variance
  - Far smaller classifier with faster training and inference time

The bottleneck of training and inference time lies in FISTA

<table>
<thead>
<tr>
<th>Model</th>
<th># Parameters</th>
<th>Training Time (s)</th>
<th>Inference Time (ms)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>FISTA</td>
<td>Total</td>
</tr>
<tr>
<td>CA-N (d=100)</td>
<td>$M \times d + d \times N_c$ (13k)</td>
<td>452.56</td>
<td>306.33</td>
<td>26.94</td>
</tr>
<tr>
<td>CA-E (d=50)</td>
<td>$r \times d + d \times N_c$ (4.25k)</td>
<td>107.15</td>
<td>61.59</td>
<td>3.50</td>
</tr>
</tbody>
</table>

$M = 128$, $r = 83$ and $N_r = 0.6$
Detailed Timing Analysis

- The bottleneck in FISTA is $\nabla_{\alpha} f$ operation
  - $\nabla_{\alpha} f = \nabla_{\alpha} \frac{1}{2} \| x - D\alpha \|_2^2 = D^T D\alpha - D^T x \to O(d^2)$
  - Above order matches the following table

- CA-E accelerates FISTA by
  - Cut off the complexity of each iteration
  - Reducing the number of iteration

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Time (s)</th>
<th>Inference Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>FISTA</td>
</tr>
<tr>
<td>CA-N (d=100)</td>
<td>452.56</td>
<td>306.33</td>
</tr>
<tr>
<td>CA-E (d=50)</td>
<td>107.15</td>
<td>61.59</td>
</tr>
</tbody>
</table>
Reference
