Computational Analysis of Neutron Scattering Data

PhD Dissertation Defense
Benjamin Martin
July 14 2015
About Me

- B.S. Computer Engineering 2009
- M.S. Computer Engineering 2012
- Intern at ORNL for 5 years
  - Worked on satellite image processing using machine learning for most of ORNL internship
- Some of my more recent research has involved data processing for neutron scattering experiments
  - Shared many similarities with my satellite imagery work
  - Focus on crystal defect detection
  - Joint effort between some of the computational groups at ORNL and groups at SNS
Quick Recap from Proposal
Crystal Structures

- Crystals are repeating structures of “unit cells” of atoms
  - Atoms are the same for all cells
  - Repeating structure is called “long-range order”
- A defect occurs when the periodic structure is disrupted
  - These defects affect material strength, thermal conductivity, pharmaceutical properties, and more.
Neutron Scattering Background

- Looking at diffuse neutron scattering
  - Used for analysis of crystal lattice structures
  - Neutrons pass through sample and create diffraction patterns
  - Diffraction patterns create reciprocal space image
    - Discrete Fourier transform for cell structure factors

![Diagram of neutron scattering process]
Neutron Scattering Background

- Two parts of reciprocal space images:
  - Bragg peaks
    - High-intensity diffraction patterns
    - Describe average crystal structure
  - Diffuse scattering
    - Low-intensity diffraction patterns
    - Describe deviations from average crystal structure
- **Goal:** Analyze textures in the reciprocal space imagery to identify defects in simulated crystal structures
  - Single crystal neutron scattering
  - Diffuse scattering patterns will be the primary focus as they describe deviations from the average crystal structure
Neutron Scattering Background

- Different defects create different diffraction patterns
- Can be viewed as a “fingerprint” for the defect
Goal: Automatically detect defects in simple simulated crystal structures for single crystal scattering experiments

General Approach:
- Extract texture features from reciprocal space images
- Look at problem as a generic data classification problem
- Minimal knowledge of underlying crystal structure needed
- No need for system changes if crystal structure changes
Preliminary Work from Proposal

• Experimental results:
  • 2-class defect classification accuracy: 98.05%
  • 3-class defect classification accuracy: 76.12%
    • Lower accuracy due to similarities between substitution classes

• Extra proof of concept work since proposal
  • Increasing class separation margin for substitutions had little to no effect on classification accuracy in 3-class problem
  • System was able to also detect substitution location
    • 64-class substitution location accuracy: 95.67%

• Random forests were found to perform better than SVMs
  • Both in accuracy and computational complexity

• Details for this preliminary work are available in dissertation
Large Structure Analysis
Overview

- Preliminary work was a proof of concept
  - Tested if defect detection methodology works at all
  - Dataset was for a toy problem
  - Crystal structure was not realistic
  - Defects were very, very simplistic
- Next step: Scale up to a larger structure
  - Defects can be more complex
  - Larger reciprocal space image size
  - Intensity range is much larger than small structure data range
Large Structure Data Properties

- Data is for close-packed crystal structures
- Simulated using the DISCUS simulator
  - Developed by Los Alamos National Laboratory
  - Uses similar methodology to (Butler and Welberry, 1992)
  - Adds extra variables to make simulation more realistic
- Crystal structure is a 100 cell by 100 cell silicon lattice
- Image size is 501 pixels by 501 pixels
  - Single-band intensity maps
- Comparison to preliminary data:
  - Lattice was 8 cells by 8 cells
  - Image size was 129 pixels by 129 pixels
Close-Packed Crystal Structures

- Close-packed crystal structures are created by stacking layers of atoms to form a crystal lattice
  - Layers denoted as letters (A, B, C, etc.)
  - Stacks are represented by strings (ABC)
- Two stacking configurations:
  - **Cubic close packed (CCP)**
    - 3-layer configuration
  - **Hexagonal close packed (HCP)**
    - 2-layer configuration
Close-Packed Structure Defects

- Two types of defects considered
  - Stacking faults
    - Switching from cubic to hexagonal structure (or vice-versa)
  - Short-range order (SRO)
    - Small areas of disorder within the crystal
Close-Packed Structure Defects

- Defects can be similar in appearance
Close-Packed Structure Defects

- Defects can be similar in appearance
Image Feature Extraction

- Keypoint features
  - Automatically detect keypoints (regions of interest) within the image and generate a descriptor for each keypoint location
  - Descriptor is feature vector describing the texture of the image at the keypoint location

\[
f = [84, 41, 21, 36, 44, 21, \ldots]\]
Image Keypoint Extractors

- 3 keypoint extraction algorithms evaluated:
  - SIFT
    - 128-dimensional feature vectors
    - Advertised benefits: “Gold standard” for keypoint features
  - SURF
    - Similar to SIFT, slightly different features (approximations)
    - 64-dimensional feature vectors
    - Advertised benefits: Faster than SIFT
  - ORB
    - Open-source alternative to SIFT and SURF
    - 256-dimensional binary feature vectors
    - Advertised benefits: Real-time performance, high noise robustness
Defect Detection Methodology

- Two challenges were posed by the new data:
  - Large image intensity range
  - Increased volume of detected keypoints due to larger image size
- In order to accommodate for the large range, a preprocessing step was added that scales the data before keypoint extraction
  - Improved keypoint detection for diffuse textures
- The increased number of detected keypoints was addressed by training on only 10% of the keypoints for each image
  - Reduced time required to train classifier without significantly affecting accuracy
Defect Detection Methodology

• Two challenges were posed by the new data:
  • Large image intensity range
  • Increased volume of detected keypoints due to larger image size

• In order to accommodate for the large range, a preprocessing step was added that scales the data before keypoint extraction
  • Improved keypoint detection for diffuse textures

• The increased number of detected keypoints was addressed by training on only 10% of the keypoints for each image
  • Reduced time required to train classifier without significantly affecting accuracy
Image Preprocessing

- Large structure data intensity range is huge
  - Typically in the ballpark of [0, 10^6]
  - Range for preliminary data was approximately [0, 650]
- **Problem**: Causes problems during keypoint extraction
  - Makes keypoint detection difficult
  - Scaling is needed as a preprocessing step
- Common practice seems to be thresholding intensities at 10%–15% of the maximum intensity value
  - Percentage seems to be “eyeballed”
  - Still not good enough for keypoint extraction
Image Preprocessing

- The large data range was due to the Bragg peaks
- **Goal:** Reduce Bragg peak intensity without affecting diffuse scattering patterns
- GUI developed to assist with scaling scheme for Bragg peaks
- **Result:** Scaling methodology developed that thresholds the intensity \( I(p) \) at pixel \( p \) in the image such that:

\[
I_{\text{new}}(p) = \min(I(p), t)
\]

where threshold \( t \) is the mean intensity for the image
Image Preprocessing

• GUI Screenshot (Intensity Mode)
Image Preprocessing

- GUI Screenshot (Keypoint Mode)
Image Preprocessing

- Fixed Percentage Scaling (1% max)
Image Preprocessing

- Mean Scaling
Large Structure Experiment

- **Goal:** Classify image as belonging to 1 of 3 defect classes:
  - “No Defect”, “Stacking Fault”, “SRO”
  - Classes suggested by neutron scientists as hard to distinguish visually
- 600 images simulated via DISCUS
  - 200 No Defect (100 CCP/100 HCP)
  - 200 Stacking Fault (100 CCP/100 HCP)
  - 200 SRO (100 CCP/100 HCP)
- **Note:** No distinction was made between CCP and HCP samples during training
  - Learning to ignore stacking configuration and just focus on the defects was left to the learning algorithm
Large Structure Experiment

- **Preprocessing:**
  - Images scaled via mean scaling method
  - Linear scaling to $[0, 255]$ then performed as required by keypoint extractors
- **3 keypoint extractors tested:** SIFT, SURF, and ORB
- **Training:**
  - Random forest classifier
  - Used 10% of the images in the dataset
  - Random 10% of the keypoints in each image used for training
- **Keypoint voting used to classify test images**
- **Results averaged over 100 independent experiments**
Large Structure Experiment

- **Results:**

<table>
<thead>
<tr>
<th>Keypoint Extractor</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT</td>
<td>96.36%</td>
</tr>
<tr>
<td>SURF</td>
<td>93.04%</td>
</tr>
<tr>
<td>ORB</td>
<td>92.59%</td>
</tr>
</tbody>
</table>

- **Conclusions:**
  - This “difficult” defect detection problem was rather easy to solve using the computational defect detection methodology.
  - SIFT had highest accuracy of the keypoint extractors.
    - More on keypoint extractor evaluation in a moment…
Prediction Evaluation Criteria

- **Question:** How to evaluate the quality of a prediction?
  - What happens if there is a voting tie or general uncertainty?
- **Goal:** To reduce need for human evaluation
  - Cannot expect classifier to be perfect
  - A heuristic may be misleading
- **Solution:** Assign confidence measure to each prediction
  - Defined as the percentage of keypoints that belong to the class that “won” the vote
  - Samples with confidence falling below a predefined threshold can be flagged for human evaluation
Prediction Evaluation Criteria

- Mean confidence for experiment

<table>
<thead>
<tr>
<th>Keypoint Extractor</th>
<th>Mean Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT</td>
<td>75.98%</td>
</tr>
<tr>
<td>SURF</td>
<td>81.61%</td>
</tr>
<tr>
<td>ORB</td>
<td>79.39%</td>
</tr>
</tbody>
</table>

- **Word of Caution:** A high mean confidence does not imply high accuracy
  - Primary goal is to maximize accuracy
  - Only then can confidence be maximized
The keypoint extractors were evaluated using two criteria:

- Classification accuracy
- Computational complexity with respect to image size

Classification accuracy
- SIFT had higher accuracy than SURF or ORB

Computational complexity
- All three extractors have complexity $O(mn)$ for an image of dimensions $m$ pixels by $n$ pixels
  - Detailed ORB analysis is available in dissertation appendix
- However, there is more to consider…
Keypoint Extractor Evaluation

- Benchmark graph for keypoint extractors:
Keypoint Extractor Evaluation

- Computational complexity observations:
  - Computational complexities are the same, but the running times are very different
  - Times required to process a single image vary by algorithm
  - Longer feature vectors cause subsequent processing steps to require more time to complete

- Summary:
  - SIFT has higher accuracy at the cost of longer running times
  - ORB runs faster than SIFT at the cost of lower accuracy
  - A researcher will need to consider the tradeoff between higher accuracy and shorter completion time
Conclusion
Conclusion

- Crystal defects can be detected using image processing and machine learning methods
  - Detection methodology presented and verified using a series of increasingly difficult problems
  - Scaling methodology developed to handle large intensity ranges
  - Method to handle larger image sizes also evaluated
- Random forests most effective in detecting defects
- SIFT and ORB were the top performing keypoint extractors
- Confidence measure can be used to address uncertainty
Future Work

- Real data analysis
  - What modifications will need to be made when using real data?
- Experimentation with multiple defects
  - Is it possible to detect two different defect types in an image?
- Defect texture analysis
  - What textures are unique to a specific type of defect?
    - Could help with classifying subtle differences
- Sensitivity quantification
  - How subtle must defects be before they cannot be detected?
    - First step: Determine which types of defects are hardest to detect
    - Does sensitivity change across periodic table?
- Future publication expected through ORNL/SNS
Summary of Contributions

- Evaluation of data processing methodologies for scattering data
- Analysis of reciprocal space imagery characteristics
- Development of scaling methodology for scattering data
- Creation of GUI to aid in reciprocal space analysis
- Formalization of defect detection methodology evaluated using following test cases
  - Classification of simple defect types in small structures
  - Prediction of defect properties in small structures
  - Detection of more complex faults in larger structures
- Comparison of keypoint extractor and machine learner performance in the context of reciprocal space imagery
  - Including detailed complexity analysis for ORB keypoint extractor
Goals from Proposal

- All goals from proposal completed
  - Small structures: Analysis of substitution class separation
  - Small structures: Detection of substitution location
  - Large structures: Analysis of data properties
  - Development of scaling methodology
  - Defect detection for large crystal structures
  - Evaluation of feature extractors and machine learning methods
    - Including computational complexity analysis
    - Detailed analysis for ORB
  - Study of tie-breaking and confidence for defect predictions
Thank You

Questions?
Extra Slides
Current Detection Methodology

- State-of-the-art crystal defect detection:
Current Detection Methodology

- State-of-the-art crystal defect detection:

**Sample 1000**

**Sample 0**

**SPOT THE DIFFERENCE**
Current Detection Methodology

- State-of-the-art crystal defect detection:

SPOT THE DIFFERENCE
(HINT: HERE’S A DIFF)
Sample Reciprocal Space Image
Reciprocal Space Definition

- Total complex scattered amplitude:
  - $A(k) = \sum_{m=1}^{N} F_m \exp(i k \cdot R_m)$ where:
    - $N = \text{number of cells in the lattice}$
    - $F_m = \text{structure factor for } m^{th} \text{ cell (listed below)}$
    - $k = \text{diffraction wave vector}$
    - $R_m = \text{position vector of } m^{th} \text{ cell}$

- Structure factor:
  - $F_m = \sum_{n=1}^{N_m} f_n \exp(i k \cdot r_n)$ where:
    - $f_n = \text{scattering factor for atom } n$
    - $r_n = \text{location of atom } n \text{ within the cell}$

- Reciprocal space intensity at $k$:
  - $I(k) = A(k)A^*(k)$
  - Reciprocal space images are basically the DFT magnitude for the structure
  - Phase problem: Phase data lost = Unable to do inverse transform
Feature Extraction Example

Sample 2

Sample 2 - 46 keypoints
Data Information

- Toy dataset
  - 8 cell by 8 cell crystal lattice
  - 129 pixel by 129 pixel intensity maps
  - Cells contain two atoms with different scattering factors
  - Crystal is for proof of concept
    - Not intended to represent a realistic crystal

- Reciprocal space images: Single band pixel intensity maps

- Simulated dataset
  - Generated with the help of ORNL staff using methodology presented in (Butler and Welberry, 1992)
  - Simulations are apparently very accurate and seem to be a common step before performing neutron scattering experiment
Feature Classification

- Any classifier can be used at this point
- Three types of classifiers were evaluated in the experiments:
  - Support vector machine (Linear kernel)
  - Support vector machine (RBF kernel)
  - Random forest
- Input data points:
  - Keypoint descriptors
  - Corresponding label for the image they were extracted from
- Classification of a new image involves:
  - Collecting predictions for all of the keypoints in the image
  - Assigning a final label via a majority vote of the keypoints
Support Vector Machines

- SVMs seek to create a decision boundary that maximizes the margin between two classes
- They are a standard baseline method
- A kernel functions can be used to aid in separation
  - Linear and radial basis function (RBF) evaluated
Random Forests

- Random forests are ensembles of decision trees
  - Each tree uses a different subset of the data
  - Each tree node uses a subset of features to make decision
  - Final classification is via vote or average of tree classifications
Comparison of Learning Algorithms

- Learning algorithms evaluated using two criteria:
  - Classification accuracy
  - Computational complexity with respect to training sample volume

- Classification accuracy
  - Random forests had consistently higher classification accuracy

- Computational complexity for $N$ training samples
  - SVM training: $O(N^2) – O(N^3)$
  - Random forest training: $O(N \times \log(N))$

- **Conclusion:** Random forest is the better choice
  - It had higher accuracy in the experiments
  - It has lower computational complexity for training
Experiment: 2-class Problem

- **Goal:** Classify a crystal containing one of two defect types:
  - Substitution (small and large)
    - Small - scattering factor on [0,1]
    - Large - scattering factor on (1,2)
  - Shear

- Simple problem to evaluate the effectiveness of the proposed defect detection methodology
Experiment: 2-class Problem

- 600 images
  - 400 substitution (200 large, 200 small)
  - 200 shear
- SIFT descriptors extracted from each image
  - Extractor requires images to be scaled to range $[0,255]$
- Training procedure:
  - 3 learners tested: SVM (linear), SVM (RBF), and random forest
  - Learner trained using keypoint descriptors
    - Trained on 10% of images
    - Image label is assigned to each keypoint
- Class of test image determined via majority vote of its keypoints
- Results averaged over 20 independent experiments
Experiment: 2-class Problem

• **Results:**

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM (linear)</td>
<td>97.31%</td>
</tr>
<tr>
<td>SVM (RBF)</td>
<td>95.92%</td>
</tr>
<tr>
<td>Random Forest</td>
<td>98.05%</td>
</tr>
</tbody>
</table>

• **Conclusion:**
  • Methodology does good job of detecting defects
  • All classifiers performed very well in this experiment

• **Next step:** Test using a more difficult problem
Experiment: 3-class Problem

- **Goal:** Present harder problem to classifier to test the sensitivity of the defect detection methodology
- Split substitution class into “large substitution” and “small substitution” subsets
  - Harder to distinguish between these classes
- 600 images
  - 200 large substitution
  - 200 small substitution
  - 200 shear
- Training and classification procedure was the same as the previous 2-class experiment
Experiment: 3-class Problem

• **Results:**

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM (linear)</td>
<td>70.87%</td>
</tr>
<tr>
<td>SVM (RBF)</td>
<td>70.56%</td>
</tr>
<tr>
<td>Random Forest</td>
<td>76.12%</td>
</tr>
</tbody>
</table>

• **Conclusions:**
  • Methodology is precise enough to predict subtle defect differences
  • Random forest performed much better than the SVMs
  • Lower overall accuracy was due to confusion between large and small substitution classes
    • Increasing class separation did not significantly affect results
Experiment: Substitution Location

- **Goal:** Evaluate whether classification methodology can be used to detect other specific properties of a defect
  - Can location of substitution be predicted?
- 1000 large substitution images
  - Substitution can be in 1 of 64 possible cell locations
- Feature extraction and machine learning set-up was the same as the other defect classification experiments
  - Classification label is the integer index [0,63] for the cell containing the substitution defect
Experiment: Substitution Location

- **Results:**

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM (linear)</td>
<td>94.80%</td>
</tr>
<tr>
<td>SVM (RBF)</td>
<td>73.76%</td>
</tr>
<tr>
<td>Random Forest</td>
<td>95.67%</td>
</tr>
</tbody>
</table>

- **Conclusions:**
  - It is possible to predict specific defect properties
  - Random forest and linear SVM performed very well
  - SVM with RBF kernel did not perform as well