Project Checkpoint #2

Kristen Vinh
kv1895@rit.edu

I. RECENT PROGRESS

A. *Hierarchy Adjustments for Classification Algorithms*

During the early stages of testing classification models on the sweater construction tags (circular yoke, raglan sleeves, drop sleeves, set-in sleeves, dolman sleeves, saddle shoulders), I discovered that the "sweater construction" category I created as an amalgamation across various tag categories likely wasn't going to work – many of the tags had a zero or very, very low scores across precision, recall and F1-Scores. In addition, none of the models I ran for sweater construction had a test accuracy above 35 percent. Since this was a key part of the upper levels of my hierarchy, I adjusted to incorporate the attributes differently: in level 2: Task C (circular yoke) and level 3: Task D (all others, which were sleeve features).
The final hierarchy I developed was as follows:

(Level 2 Task A has been struck through here, as it is part I tested, but I eventually decided to move it into level 3 Task B as part of the sleeve length of the level 3hierarchy – see level 2 Algorithm Testing and Benchmarking)

Level 1: Base Sweater Type: Binary Classification

- Task: Cardigan or Pullover: Classify the image as either a **Pullover** or a **Cardigan**.

Level 2: Primary Structural Features: Binary Classification

These are also binary classifications, and I'll test both across the entire database and separately on pullovers and cardigans to determine whether the hierarchy is necessary.

- ~~**Task A: Has Sleeves or Sleeveless:** See if an algorithm can accurately label the sleeves using the "attr_sleeveless" tag.~~
- **Task B: Colorwork Type: Single Color vs. Multi-Color**.
- **Task C: Circular Yoke or not:** This is the only primary tag that doesn't fit elsewhere, but it is a widespread sweater construction, so it feels important to include. Also, it's mostly pullovers (although some cardigans), so it might benefit from being its own task.

Level 3: Secondary Detail Features: Multilabel Classification

These models focus on more granular details that may depend on the above (i.e., it must be multicolor to feature a type of colorwork and/or are beyond binary classification).

- **Task A: Neck Type:** Classify the neckline (e.g., *Crew, V-neck, Turtleneck, Shawl Collar*). Note: combines turtleneck and mock turtleneck sleeve, since an algorithm likely wouldn't be able to tell the difference.
- **Task B: Sleeve Length:** Classify sleeve length (*Long, Three-quarter, Short*)
- **Task C: Type of colorwork:** (stripes, corrugated ribbing, intarsia, etc.)
- **Task D: Sleeve features**: (*Puff, Bell*, etc.). Note: combines drop and modified drop sleeve, since an algorithm likely wouldn't be able to tell the difference.

Since these algorithms were taking quite a long time to run, I decided to test each of the four classification algorithms (LetNet5, AlexNet, ResNet50, VGG-19) on sweater type classification (cardigan versus pullover), optimize those models, and proceed with the best-performing model to level 2 classification. If needed, I would perform further testing on the second-best-performing model. I chose these algorithms because they are known for performing well at classifying images. I planned to test these in order of increasing complexity, starting with LetNet5, AlexNet, ResNet50, and then finally, VGG-19. The latter two, ResNet50 and VGG-19, are pre-trained on the ImageNet dataset and are more advanced, so I did them last.

B. *Algorithm Benchmarking and Evaluation Planning*

a.) Classification Algorithm Benchmarking: In this phase of the project, I began to determine ways to benchmark each of these algorithms. Given the hierarchical nature of the classification algorithms, I planned to evaluate individual models using three metrics: precision, recall, and F1 score. I intended to get scores for each of the four models on the level 1 classification task (pullovers versus cardigans), then do the level 2 and level 3 testing using the best-performing model from level 1. I also planned to utilize Grad-CAM to examine the features used to classify correct and misclassified images.

b.) *ANN Algorithm Benchmarking:* To evaluate the ANNOY algorithm (and other ANNs), I will use the following metrics: indexing time (the time required to build each index), recall (accuracy comparing the actual nearest neighbors using Scikit-learn's Nearest Neighbors vs. the ANN's nearest neighbors), and index size (the final index size on disk). For the best-performing algorithm, I will also test the "in-the-wild" data from Reddit. I will evaluate the performance of the "in-the-wild" data by using qualitative scores (see Table I): Does it have the same pattern type (cardigan vs. pullover), similar construction (same sleeves, neck, collar), and the same attributes (colorwork, cables, lacework, etc.)? This is similar to the hierarchy used in classification, but with additional attributes. Here, metrics like circular yoke, sleeve, and extra features are weighted more heavily because they are harder to distinguish on and are more difficult for a knitter to adjust.

Another way I may test is by downloading sweaters knit from the patterns in my dataset and scoring them based on whether the pattern is among the five recommendations. Both of these evaluation methods are time-consuming. While the second method (testing sweaters knit from patterns in my dataset) is less subjective, given that Ravelry has a lot of similar sweater patterns, this method won't account for the algorithm finding a close match that is NOT the actual pattern but matches a lot of the characteristics (which, as you'll see below, happens frequently).

Finally, I will also develop tests using Grad-CAM to display the features used to find the nearest-neighbor pairs of sweaters.

TABLE I. QUALITATIVE SCORECARD FOR ANN EVALUATION

Metric	**Description**	**Points**
Type	Do the recommendations match the sweater type of the pattern (cardigan or pullover)?	5
Number of Colors	Do the recommendations match the number of colors the pattern has (one or multiple)?	5
Sleeve Length	Do the recommendations match the sleeve length?	5
Neck Type	Do the recommendations match the neck type (v-neck, crew, shawl, etc.) of the pattern?	5
Circular Yoke	Do the recommendations match the pattern's yoke type (circular or other yoke type)?	10
Sleeve Features	Do the recommendations have similar sleeve features (raglan, drop, bell) to the pattern?	10
Additional Features	Do the recommendations share features with the pattern, such as stockinette stitch, colorwork, cablework, lacework, etc.?	10
Total Points		50

C. *Level 1 Algorithm Testing and Benchmarking*

Because the classification models were taking a long time to run, I wanted to narrow it down to the top-performing algorithm before moving on to the next level of my hierarchy. The classification scores for each model at the level 1 hierarchy are presented in Tables II, III, IV, and V.

TABLE II. COMPARISON OF PRECISION SCORES FOR VARIOUS MODELS, LEVEL 1 HIERARCHY CLASSIFICATION

	LeNet5	**VGG-19**	**ResNet50**	**AlexNet**
pullover	0.58	0.82	0.82	0.66
cardigan	0.68	0.72	0.84	0.70
macro avg	0.63	0.77	0.83	0.68
weighted avg	0.64	0.78	0.83	0.68

TABLE III. COMPARISON OF RECALL SCORES FOR VARIOUS MODELS, LEVEL 1 HIERARCHY CLASSIFICATION

	LeNet5	**VGG-19**	**ResNet50**	**AlexNet**
pullover	0.44	0.81	0.74	0.44
cardigan	0.79	0.73	0.89	0.85
macro avg	0.62	0.77	0.82	0.64
weighted avg	0.65	0.78	0.83	0.69

TABLE IV. COMPARISON OF F-1 SCORES FOR VARIOUS MODELS, LEVEL 1 HIERARCHY CLASSIFICATION

	LeNet5	**VGG-19**	**ResNet50**	**AlexNet**
pullover	0.50	0.81	0.78	0.53
cardigan	0.73	0.72	0.86	0.76
macro avg	0.62	0.77	0.82	0.65
weighted avg	0.64	0.78	0.83	0.67

TABLE V. COMPARISON OF OVERALL ACCURACY SCORES FOR VARIOUS MODELS, LEVEL 1 HIERARCHY CLASSIFICATION

Model	**Accuracy**
LeNet5	0.65
VGG-19	0.78
ResNet50	0.83
AlexNet	0.69

ResNet50 achieved the best scores across all metrics, and VGG-19 ranked second. Due to time constraints, I decided to focus on creating level 2 classification models using ResNet50 for this part of the project.

D. Level 2 Algorithm Testing and Benchmarking

Before I began the level 2 hierarchy testing, I needed to download more samples of circular yokes and sleeveless sweaters. Many of these sweaters were not as well tagged as the initial batch of samples, but I decided to include them to see if I could improve models for classes with low sample counts.

Then, for the color and circular-yoke levels of the level 2 hierarchy, I tested whether I got a better weighted average by running the Resnet50 algorithm using a hierarchy (separately on cardigans and pullovers) versus all sweaters together:

- Colorwork hierarchy models had a weighted F1-Score of 0.79 (with pullovers = 0.79 and cardigans = 0.80)
    - When run on all sweaters together, the F1 Score was 0.79
- Circular yoke hierarchy models had a weighted F-1 Score of = 0.87 (with pullovers = 0.85 and cardigans = 0.90),
    - When running the circular yoke classification on all sweaters, the F1 Score was 0.85.

These are close, and given that some classes had really low sample sizes (circular yoke cardigans, for example), I've decided to stick with the non-hierarchical models for now, even though pullover models sometimes performed better than the overall class — this will allow for greater simplicity and accuracy overall.

Despite the sample boosting, predicting sleeveless vs. sleeves continued to fail with more samples, having low precision, recall, and F1-score for the sleeveless class, as the algorithm tended to default to "has sleeves" to achieve high accuracy (see Table VI). As noted above, I've decided to remove this from my algorithm at this stage and see if it performs better as part of sleeve length.

E. Algorithm Testing and Benchmarking on Level 3

As I began classifying on the finer details in level 3, precision, recall, and F1-score were low. In my classification test of neck styles, running a preliminary test on only the top five neck styles yielded 44% test accuracy, with low precision, recall, and F1 scores. Similarly low test accuracies were obtained with the running sleeve length (63%), colorwork (49%), and sleeve feature (36%) algorithms.

The last algorithm benchmarking on level 3 models performed poorly, so using classification algorithms was unlikely to yield good results overall. Therefore, I chose not to further evaluate these models in the end-to-end method I had begun to develop. Depending on how the approximate nearest neighbors (ANNs) algorithms perform, I may test level 2 and level 3 hierarchies on VGG-19.

F. Real-Life Example Processing

This project phase also included downloading real-life examples from Reddit for final testing. After using Reddit's API to find search results for individuals seeking pattern recommendations, this becomes a manual process of identifying sweater characteristics that match the classification algorithms and determining whether a Ravelry pattern suggestion was made in the comments. This process will continue over the next several weeks, as new requests are posted.

G. Shift to ANN Algorithm Testing

I also learned that, as I manually went through Reddit examples, often similar shapes, patterns, and cables are more important than type (cardigan vs. pullover), since modifications can be made. Frequently, people are also looking for things that go deeper than I've decided to go with my hierarchy — lace, cables, etc. — that my hierarchy classification isn't designed to classify on. I determined it made sense to go deeper into testing and evaluating ANN algorithms beyond ANNOY, including FAISS, HNSWlib, and Voyager.

ANNs are a potentially good solution for this task because they tend to find similar items in large datasets very quickly and use a machine learning algorithm (I plan to use ResNet50, my best-performing model from above) to extract features and store them as vectors for quick similarity matching.

I started with ANNOY because it was easy to implement and straightforward—while it's not the most advanced (Spotify has since released a newer ANN), it's a good baseline. Next, I'll move on to Hierarchical Navigable Small World (HNSWlib), which is considered to be state-of-the-art for high-performance, high-accuracy ANN searches. Then, I'll move on to Voyager (Spotify's ANNOY successor), which is built on HNSW and will serve as a good comparison to HNSWlib. Finally, I'll test Facebook AI Similarity Search (FAISS), a library of several indexing algorithms. I'll start by testing its IndexFlatIP, which compares each sweater to every other sweater in the database using an inner product calculation and the HNSW implementation (which can be memory-intensive, but performs very well in terms of accuracy and speed).

H. Initial ANNOY Algorithm Testing

I began building my ANNOY algorithm using a similar approach to the Fashion-Recommender-System [1], replacing nearest-neighbor with ANNOY. Initial results showed some promise, but also had some issues, such as images being shown just because they had a similar pose. Figure 1 shows the test image, and Figure 2 shows the recommendations from this original run.

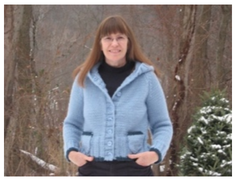

Fig. 1 Test Image for ANNOY evaluation

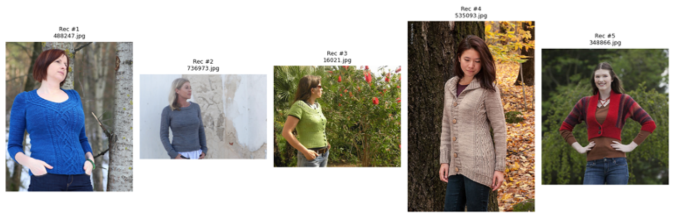

Fig 2. Recommendations from running ANNOY on the test image.

Recommendation #4 looks similar, but is not the same pattern as the sample sweater. Given the lack of accuracy in the hierarchical classification model, I will continue to explore ANNOY and other approximate nearest neighbor (ANN) models (FAISS, HNSWlib, and Voyager). In addition, I will work on improving ANNOY (and the other ANNs), using a model like YOLO for object detection to crop images to the sweater and avoid pose similarity, or using vector averaging for the ANNOY index.

I. *Testing ANNs with Vector Averaging*

I began testing the other algorithms mentioned above, writing scripts to create FAISS, HNSWlib, and Voyager indexes. Since I knew I needed a way to improve the models over my initial ANNOY run, I used vector averaging. This method pulls features using my ResNet50 algorithm and YOLO across multiple images for a given pattern. It then averages them, similar to the approach in [2], which created a prototype from the averages of all the samples. This appeared to be more effective than a single image from a pattern.

I first evaluated models based on Recall@K, index size, and build time on 10,000 sweater samples (with multiple sweaters per sample used to compute vector averages). The metrics are presented in Table VI.

TABLE VI. ANN INDEX METRIC COMPARISON

Model	Build Time	Build Size	Recall
ANNOY	10.26	150.10 MB	0.9523
FAISS-IP	0.02	77.23 MB	0.9999
FAISS-HSNW	9.06	79.79 MB	0.9836
HSNWlib	5.12	78.63 MB	0.9557
Voyager	1.36	78.33 MB	0.9255

As a result, I chose to proceed with testing example sweaters knit from patterns on Ravelry from FAISS-IP and HSNWlib. I chose them because they had smaller build sizes and higher accuracy. However, when it came to real-life example testing using samples knit from patterns in my database, the results were not as accurate as hoped, and did not match the patterns. For the most part, every model gave roughly the same recommendations, as seen in Figure 3 and Figure 4.

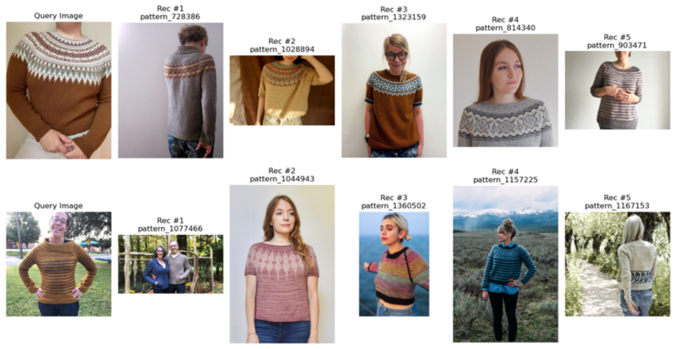

Fig. 3. FAISS-FLATIP recommendations

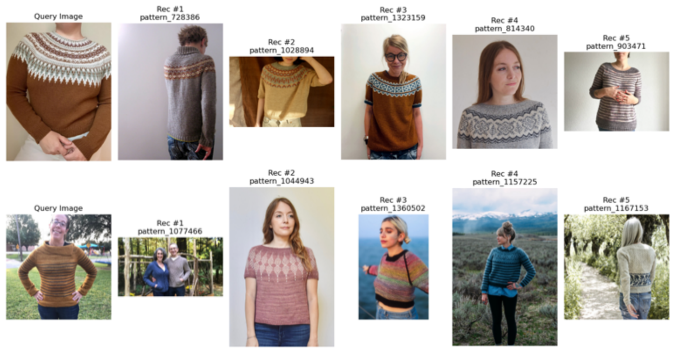

Fig. 4. HNSWlib recommendations

While the first recommendation set looks better than the first, it's still too pose-based, so I decided to pivot to YOLO testing. Since recommendations are similar between the models, I will run final qualitative testing on one or two models (or models with different feature extraction methods).

II. QUESTIONS/CONCERNS

- In general, the classification models using Ravelry's tags are not meeting the standards I would prefer, so a change of course to focus on ANN algorithms is necessary. Is it OK to shift entirely to that and abandon further evaluation of classification? I'm concerned about time here.

- I'm also concerned that none of the algorithms will perform well—is that OK for my final paper and presentation?

- For the Level 2 classification tasks (colorwork and circular yoke), the performance difference between a hierarchical classification approach (e.g., training separate models for pullovers and cardigans) and a single model trained on all sweaters was relatively small, as noted above. I've opted for the more straightforward, non-hierarchical approach for this portion. There will still be some hierarchy (colorwork type depends on it being multi-color). Is that the right decision to make if I proceed with a hierarchical classification model after ANN testing?
- What's the best way to run a qualitative analysis on the test data pulled from Reddit since it's somewhat subjective? Ideally, I would have multiple people judge it, but that's not really possible given the project's completion date of December 8. I've started outlining a process above but would appreciate advice.

III. NEXT STEPS

I expect to get through steps 1-3 before the next capstone check-in is due; step 4 is the final step in this project.

1. Continue to download testing data.
   a. To date, I have about 50 samples needed from Reddit for testing the front-end of this project.
   b. Download samples of sweaters knit from patterns for evaluation step 3 part 1 (25 samples of pullovers, 25 samples of cardigans, selected randomly).

2. Continue to investigate methods to improve ANNs, such as:
   a. Use various YOLO models to detect people, crop the sweater, and remove the background.
   b. Test different feature extraction methods, such as "CLIP," a multimodal vision-and-language model from OpenAI [3].

3. Final evaluation of top-performing ANNs: This could be on different indexes, or different versions of the indexes (YOLO object detection using ResNet50 vs CLIP), especially since the indexes all seemed to perform similarly:
   a. Get base metrics, including index size, recall, and build time.
   b. Test sweaters knit from patterns in the database and verify that the recommendations match the pattern they were knit from (similar to the test in the *ANNOY Algorithm Testing* section) or something similar.
   c. Tests on "in-the-wild" data from Reddit. These will be scored qualitatively: Does the test sample have the same pattern type (cardigan vs. pullover), construction (same sleeves, neck, collar), and the same attributes (colorwork, cables, lacework, etc.)? Likely only on the highest performing models.
   d. Apply Grad-CAM to the highest-performing models and ensure they use relevant features for recommendations.
   e. Set the final model (hierarchical or ANN) to be used in the front-end implementation.
4. Develop and implement a web interface that lets a user upload a photo and receive a series of recommended patterns from Ravelry. The web interface will integrate the machine learning model and the Ravelry API (using Ravelry's search function).

REFERENCES

[1]M. Sridevi, N. ManikyaArun, M. Sheshikala, and E. Sudarshan, "Personalized fashion recommender system with image-based neural networks," IOP Conference Series: Materials Science and Engineering, vol. 981, p. 022073, Sep. 2025, doi: https://doi.org/10.1088/1757-899x/981/2/022073
[2] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical Networks for Few-shot Learning," June 19, 2017, arXiv: arXiv:1703.05175. doi: 10.48550/arXiv.1703.05175.
[3] L. G. A. PhD, "Building an Image Similarity Search Engine with FAISS and CLIP," TDS Archive. Accessed: Oct. 27, 2025. [Online]. Available: https://medium.com/data-science/building-an-image-similarity-search-engine-with-faiss-and-clip-2211126d08fa