InterLACE Worksheet

John Spiva, Sam Heilbron, Samantha Welch

We can all add to each of the sections, but if there are any sections you would prefer to focus on / start with, just add your name to the part / question that you want to answer

1. Problem and motivation (same as design document)
2. Sponsor
    - Description of sponsor and sponsor's goals
    - How was the interaction with your sponsor?
    - What was the sponsor's assessment of your final product?
3. Original proposed solution
    - How does it solve the problem?
    - User experience
    - **Technical components and methods (Largest part)**
4. Project outcome -- what did you end up with?
    - Was it a success? A failure? Something in between?
    - What did you need to change and why?
    - How did that impact the scope of the problem/solution?
    - **How did you evaluate your solution? Describe any quantitative results.**
    - Future work: if you were to continue working on it, what would you do?
5. Lessons learned
    - Was your assessment of the risks and potential failures accurate?
    - What were some of the unexpected challenges?
    - How could the class itself be improved?
    - What advice would you give to next year's class?

Problem Motivation

The technological revolution has impacted almost every component of modern day society. From connecting everyone in the world with smartphones and a social network to

Why have technological advancements not reshaped the education system? A misconception would be to think that there are not enough resources in advancing education in the classroom. In 2015, the spending on classroom technologies worldwide reached over 15 billion dollars. *The reason these advancements haven't had a significant impact is because simply adding technology to classrooms does not inherently improve education.*

Take for example the inclusion of iPads and laptops in classrooms. Many believed this would improve education because of the ease of access to information. However, there are 4 major concerns that were overlooked:

Cost: Only a subset of schools can afford and maintain laptops for all students which increases the financial problems of the education system

Distraction: Laptops are associated more with entertainment than education

Reading: Reading comprehension is lower on screens than on paper books

Screen Time: Screens overtax attention resources and decrease social interactions

Therefore, while the inclusion of technology in the classroom is both beneficial and necessary for augmenting learning (and teaching), it is vital that implementation strategies are cognizant of unintended effects on a user base which is predominantly young and extremely malleable.

Sponsor

Our sponsors were Ethan Danahy, a Research Assistant Professor in the Tufts University Department of Computer Science and Tufts Center for Engineering Education and Outreach (CEEO), and Ean Wong, an Education Technology Specialist in the Tufts Center for Engineering Education and Outreach.

The CEEO is a leader in supporting efforts to integrate engineering into K-12 education. Its core goals are to increase excitement for learning STEM, improve teaching skills to make learning more enjoyable, increase the public's technology literacy and increase the awareness of the importance of STEM in society today.

We firmly believe not only in the mission of the CEEO, but also the implementation strategies. Therefore, we elected to work on an existing research project, led by Professor Danahy. The Interactive Learning and Collaborative Environment (InterLACE) is an NSF-sponsored research project based out of the CEEO. It is focused on promoting a collaborative classroom environment, where students negotiate and share relevant information, a design-based curriculum, where real world contexts are used to scaffold learning, and inquiry teaching, where teachers guide students through a process of exploration. Research has proven a positive correlation between collaborative, design-based, inquiry teaching and strong student conceptual gains.

We branched off the InterLACE project to create the InterLACE Worksheet. The purpose was to improve the use of worksheets in classrooms. Worksheets are great! They are tactile, require no training, and promote freedom of expression. However:

1. **Worksheets do not promote collaboration between students**
2. **Teachers do not have a definitive method for analyzing student responses and classroom trends**

Our goal was to create software to analyze the results of worksheets and display those results to the class. This would provide teachers immediate analysis of the students comprehension of material and promote collaboration between students by making them aware of each other's ideas.

Competitors

To better understand the challenge of introducing innovative technology into a classroom effectively, we researched competitors who had attempted similar solutions. Below is a brief description of some competitors:

LiveScribe: A smartpen with an embedded computer and audio recorder
Sketchboard: Team sketching on an online whiteboard
GradeCam: Grades students multiple choice answers

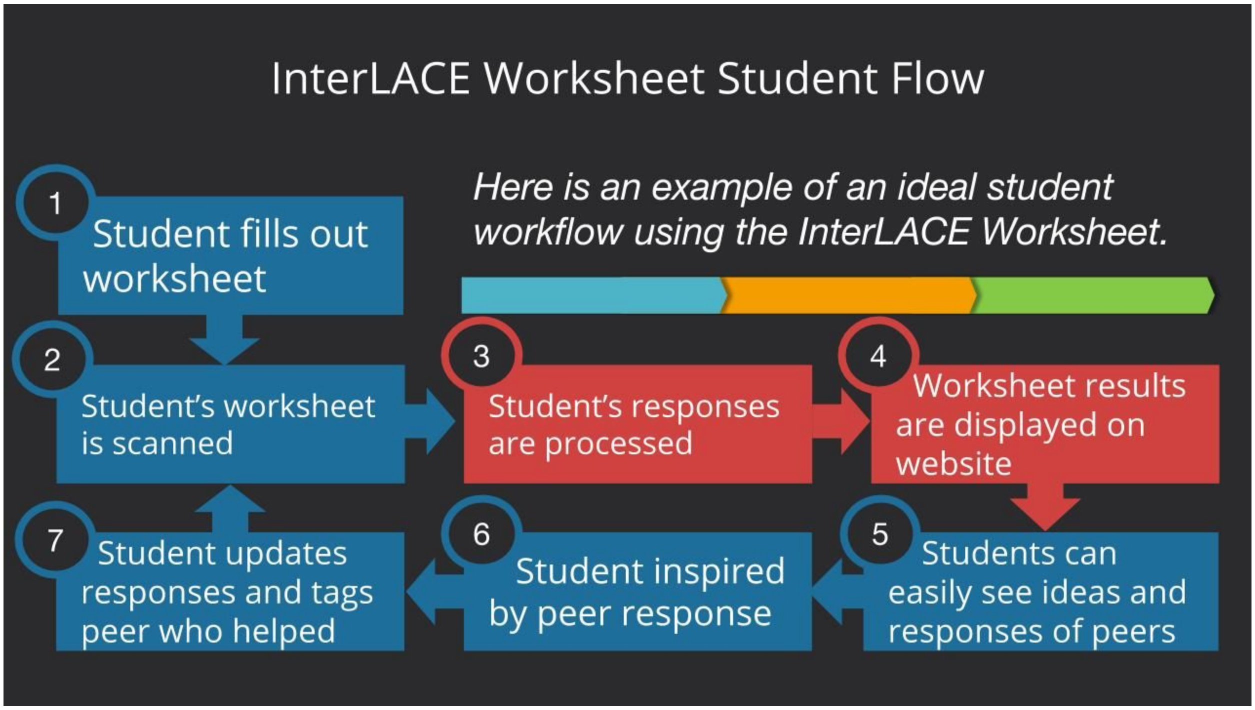
We found two major trends amongst education technologies:

1. There is a tradeoff between solutions that promote collaboration and remain intuitive to students
2. There is a positive correlation between the extent to which a solution promotes collaboration and high cost of implementing that solution.

Original Proposed Solution

The InterLACE worksheet is a low cost solution which promotes collaboration while minimally disrupting the existing classroom workflow.

Student Workflow
Below is a diagram outlining the way students will complete worksheets in their classroom while using our software. The red section of the diagram is the steps our work was focused on. Notice the cyclic nature of the workflow. Our solution is designed to promote an iterative process - students should collaborate with and question each other and then update their ideas.

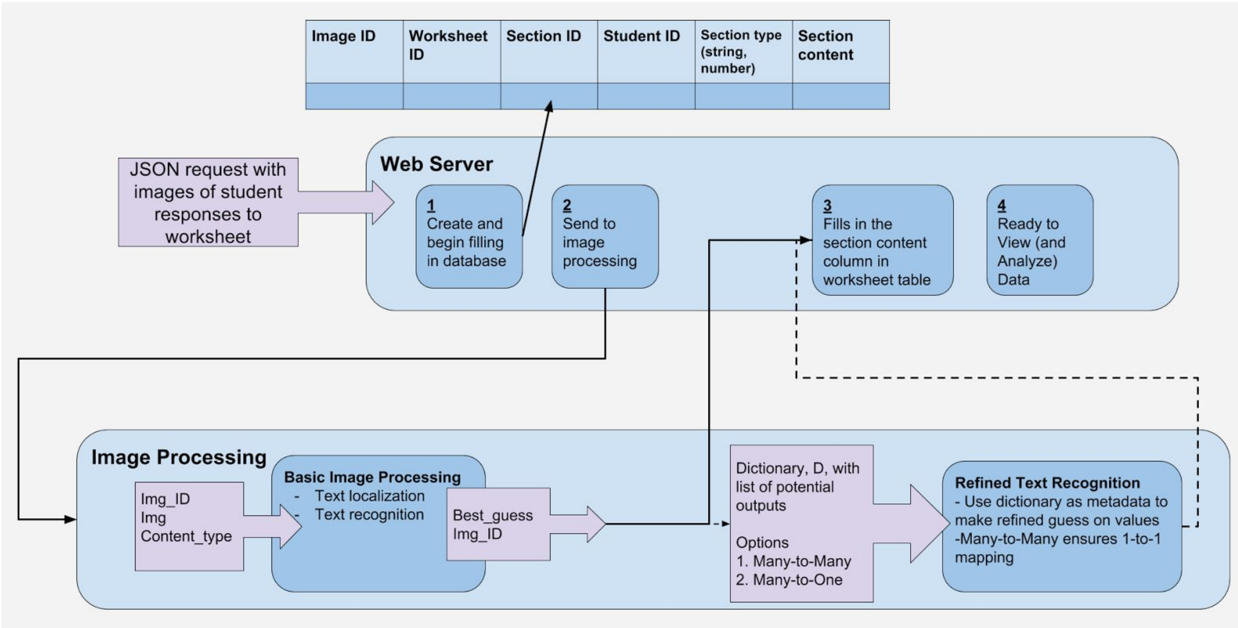


Effect on Teacher Workflow

- ***Minimally disruptive***: teachers use a batch scanner to collect worksheets and get immediate feedback on their students' work
- ***Informative visualizations*** about how their class collaborates and identify actions that encourage collaboration
- ***Immediate analysis*** of students' works means teachers can make ***data-driven decisions*** to slow down or speed up the pace of their classes

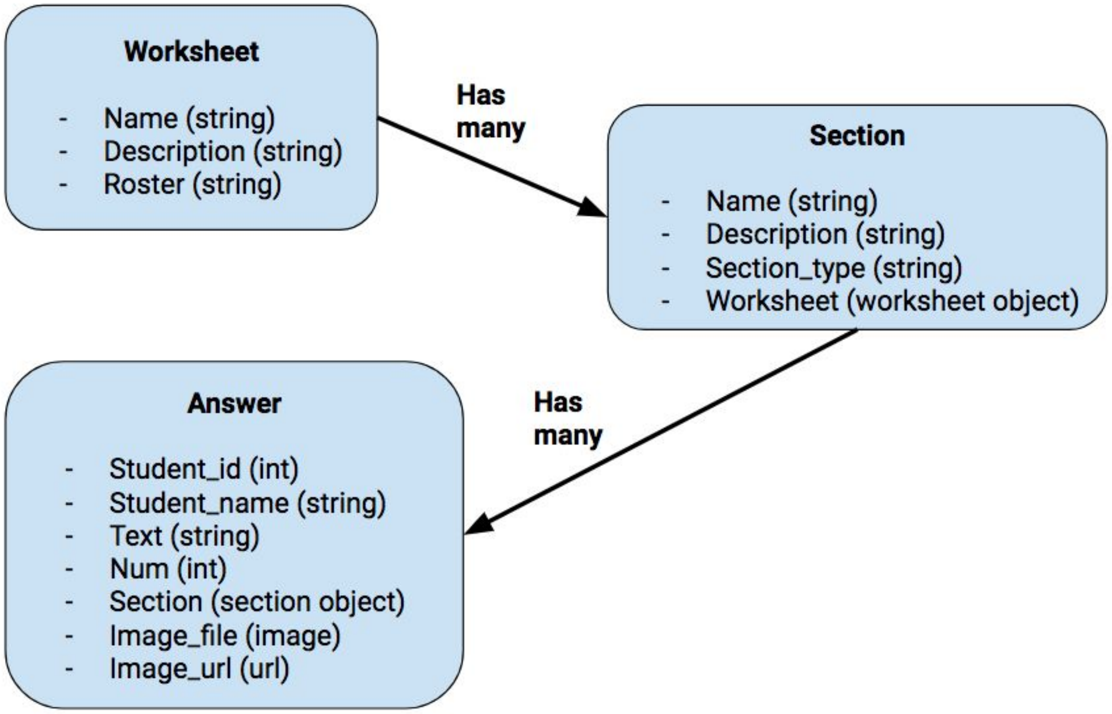
Technical Components

Our project is composed of an image processing module and a web interface. Our sponsor's primary goal was an image processing system but we wanted to add a web interface to illustrate the benefits of our solution.



Web Interface

The web interface of our project was built with the Django framework for Python. We chose Django and Python because we had already chosen to do our image processing in Python and wanted to make connecting the two as easy as possible. We set up the model in our application as follows:



The Model

This setup made it easy to retrieve all students' answers for a given sections and all sections for a given worksheet. The *section_type* field for a section allows someone adding a section object to specify the type of the content in the section so that the visualizations displayed on the website can be tailored accordingly. Currently, the types our application accepts are *Collaborators*, which displays a force-directed graph of who in the class worked with whom, *Names*, which associates each student_id in the database with the students actual name, and *Num*, which creates a bar graph of the number that each student wrote in that section.

The Visualizations

In order to create these visualizations, we utilized a few different JavaScript and jQuery frameworks and plugins. To create the dynamic data tables in our application--there is one that displays the worksheets, one that displays the sections for a worksheet, and one that displays the answers for a section, we used the *DataTables* plugin for jQuery. This plugin makes it very easy to create responsive tables that can easily be searched and reordered based on the contents of a column, and provides many other advanced features.

The worksheets data table:

Search:

Worksheet Names	Worksheet Descriptions
Worksheet 1	Short math
Worksheet 2	Negative numbers introduction
Worksheet 3	Division practice
Worksheet 4	Advanced division
Worksheet 5	Addition and division

Showing 1 to 15 of 15 entries

The sections data table:

Search:

Section Names	Section Descriptions
Names	The students who completed this worksheet
Division	33 / 11
Addition	39 + -33
Class Collaboration	Who each student worked with

Showing 0 to 0 of 0 entries

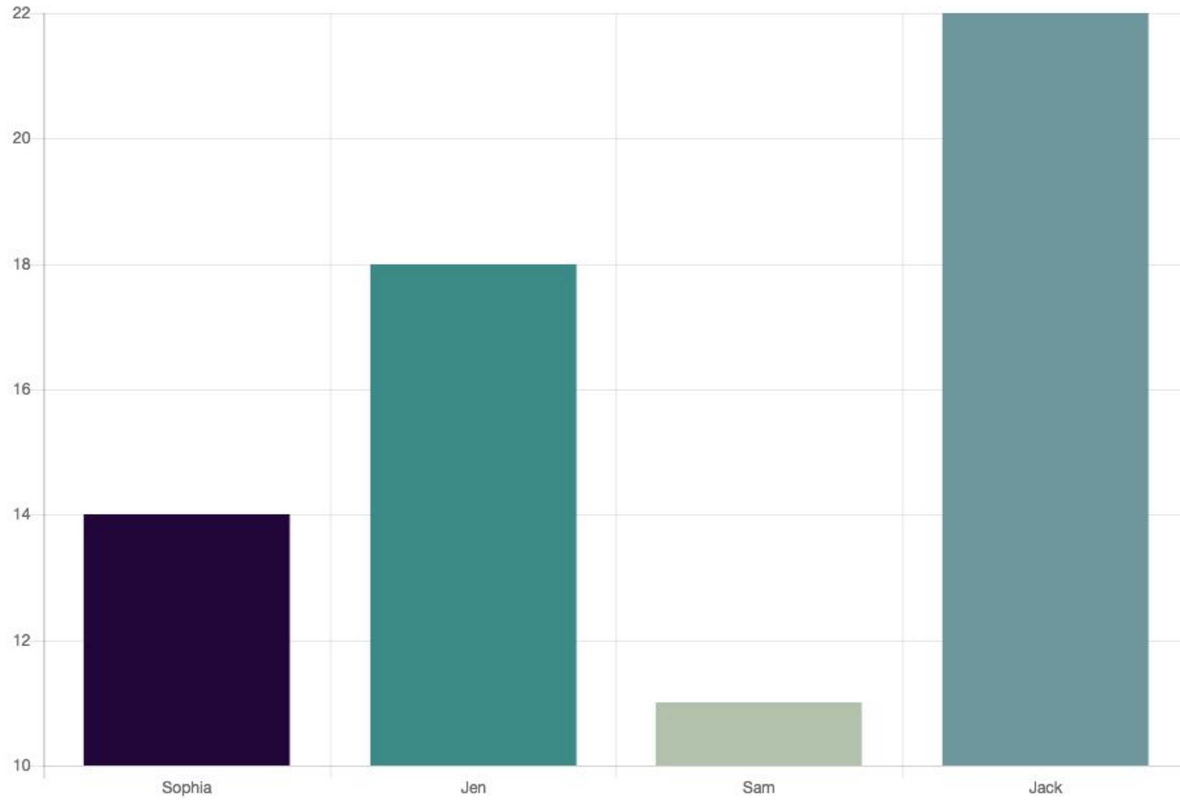
The answers data table:

Search:

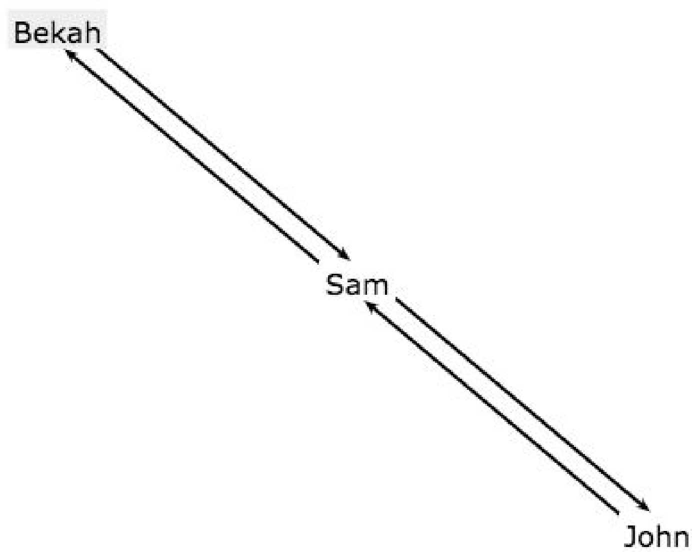
ID	Name	Image
1	Bekah	http://i.imgur.com/UEGCeeK.png
2	John	http://i.imgur.com/oENVrcW.png
3	Sam	http://i.imgur.com/7tMv02V.png

Showing 1 to 3 of 3 entries

To create the bar chart visualizing students' answers for a numerical section, we used the JavaScript Chart.js. This is a relatively simple JavaScript charting library, but met our needs well. The only issue we had with it was that it currently provides no easy way to automatically give each bar in a bar graph a different color. A single color was somewhat confusing and made it difficult to distinguish between which bar represented which student's answer, so we wrote a few simple JavaScript functions to randomly color each bar.



And finally, to create the force-directed graphs of student collaboration, we used a very simple library created by some MIT students to create force-directed graphs which provides an algorithm to calculate how to display the nodes and edges in a visually pleasing way called Springy.js. Many JavaScript libraries for creating force-directed graphs and network visualizations are quite complicated, but our needs were very basic so this one met our needs well.



The API

Creating an API for our web application was one of the most important parts of our project. The API is what allows our sponsor to connect to our application and utilize what we have built. To create the API, we used the Django REST Framework that provides a toolkit for building API endpoints for Django applications. This allowed us to easily add endpoints for creating (POST), retrieving (GET), updating (PATCH), and destroying (DELETE) answers, sections, and worksheets in our application. An example POST request to our API would look like the following:

```json
{
    "name": "Worksheet 1",
    "description": "Worksheet 1 was completed at the beginning of class on Friday.",
    "roster": "John, Sam, Bekah",
    "sections":
        [
            {
                "name": "Names",
                "section_type": "Names",
                "description": "The students who completed this worksheet",
                "answers":
                    [
                        {
                            "student_id": 1,
                            "image_url": "http://i.imgur.com/Qqyiprd.png"
                        },
                        {
                            "student_id": 2,
                            "image_url": "http://i.imgur.com/nmRwG79.png"
                        },
                        {
                            "student_id": 3,
                            "image_url": "http://i.imgur.com/CzHx5BZ.png"
                        }
                    ]
            },
            {
                "name": "Collaboration",
                "section_type": "Collaborators",
                "description": "Who they worked with.",
                "answers":
                    [
                        {
                            "student_id": 1,
                            "image_url": "http://i.imgur.com/ogKtniC.png"
                        },
                        {
                            "student_id": 2,
                            "image_url": "http://i.imgur.com/Ava3CrU.png"
                        },
                        {
                            "student_id": 3,
                            "image_url": "http://i.imgur.com/AMSOgfP.png"
                        }
                    ]
            }
        ]
}
```


Image Processing Module

Background Research

Optical Character Recognition (OCR) is the electronic conversion of images of text into machine-encoded text. There exist highly effective OCR engines which can recognize a wide range of typed fonts. However, handwritten characters pose a challenge due to the varied nature of the characters. Therefore, we began by researching custom methods for analyzing pictures of handwritten characters.

An Efficient Way of Combining SVMs for Handwritten Digit Recognition[1]**:** Hierarchical structure of SVMs following 1-against-1 approach for higher accuracy. However, instead of using n*(n-1)/2 SVMs (classifier for each permutation) proposes using n SVMs to avoid unecessary comparisons. Ensures faster processing and higher accuracy than out of the box one-against-all SVM.

Feature Detection and Tracking[2]: Chapter 9 focuses on detection of keypoints. Didn't look too closely at this. Useful resource.

New Dynamic Classifiers Selection Approach for Handwritten Recognition[3]*:* Different classifiers can offer complementary information about patterns to be classified. Good ensemble improves generalization and robustness of classification system. Subset of classifiers using local accuracy to generate output classifiers.

A MDRNN-SVM Hybrid Model for Cursive Offline Handwriting Recognition[4]: More helpful for providing list of references to other useful papers. Perhaps it will be useful to examine problems related distinguishing between characters that are connected.

The research proved to be quite helpful to better understand the steps required for image processing and text recognition. However, what became abundantly clear is that each of these solutions require large data sets to train the models. While data exists for handwritten digits, there is not much of it for handwritten alphanumeric characters. Therefore, training a custom ML solution was only effective for numbers.

Additionally, the challenging part about handwritten characters, especially that of children is that the data is highly varied, especially between different users, can be poorly spaced and oriented and may contain connected lines. Therefore, it's more feasible to approach the problem through the scope of a "Text in the Wild" problem, finding text in everyday images. These methods account for orientation, color and connected lines variations. Below are a range of papers that we read to better understand "Text in the Wild".

Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition[5]: Overviews the process of generating synthetic data and how it can be used to train effective neural nets. I still want to find more papers regarding the process because I'm still shady how the unsupervised training is managed successfully on a large feature space/data set.

Reducing the Model Order of Deep Neural Networks Using Information Theory[6]: Useful for speeding up the processing speed of neural nets without loss of prediction measure.

Deep Features for Text Spotting[7]: Haven't fully examined the paper but it's written by Max Jaderberg (now scientist at Google DeepMind) and most of the work around his thesis paper (next paper) provides the basis for our custom CNN

Reading Text in the Wild with Convolutional Neural Networks[8]: Most of the work is based off this paper. It outlines a method for pre-processing, text spotting and text recognition on two different CNN's. One is a CHARNET, which examines characters individually and one is a DICTNET, which looks at the whole word and compares it to similar dictionary words.

Text Localization

Images are preprocessed using a Gaussian blur and adaptive thresholding to remove noise. An inverted binary image is then processed. To search for individual characters, image contours are found and these bounded boxes are then processed individually. To search for full words, a sliding window is applied to the image and then a series of classification algorithms [8] determine which region of interest is the text itself.

Text Recognition

There are 3 separate methods for recognizing text. We try to use different methods depending on the metadata attached to the image to maximize correctness and processing time.

Numbers are recognized using a support vector machine (SVM), a supervised learning model which constructs a hyperplane to classify data. This is a one-against-all SVM which means that a single SVM maps to all potential outputs (0...9). The input features are the histogram of gradients (HOG) of the image.

Recognizing words proved to be extremely challenging. Originally we built a convolutional neural network (CNN), a supervised learning model whose connectivity pattern and functionality of neurons mimics axons in the biological brain. We initialized the weights based on the results of the Jaderberg paper [8]. However, since we didn't train the network explicitly, we ran into an issue related to the image size affecting the output (see future work for more information). Therefore, we explored the Microsoft Cognitive Services APIs. The APIs successfully recognize text but the processing speed and network latency make it infeasible to use live.

Refined Text Recognition

When the images are submitted, metadata about the contents of each image are attached. This could be describing the type of content in the image (number, text...etc) or the potential values of the content. Therefore, we leverage the image metadata to refine our text recognition output from the previous step.

There are two cases which we refer to as Many-to-Many and Many-to-One. Let D refer to a predefined dictionary of words (or symbols or numbers).

Many-to-One: Find the word in the dictionary with the highest probability* of being the recognized word. An example of this case is,

Many-to-Many: For each word in a list of recognized words, find the dictionary word with the highest probability* of matching. A mapping should be 1-to-1 so that no two recognized words map to the same dictionary word. Then a recursive strategy of attempting to select the most favorable available dictionary word for each recognized word is completed. This ensures that the mapping is 1-to-1 and is a simple method for attempting to maximize the total probability between all predicted words.

*We use a customized Levenshtein distance calculation for each word to determine which has the highest probability. The original Levenshtein distance only looks at mismatched characters between two strings whereas our version is customized to favor words which have successful matching characters. Our customization favors words with a few correct letters than words with none but of a similar length. For example, if our predicted word is “Tom”. The Levenshtein distance from Sam is 2 and from Tommy is 2. However, our customized distance from Sam is 1 (+2 incorrect characters -1 correct character) while Tommy is now -1 (+2 incorrect characters -3 correct characters) so we predict the student meant to write Tommy.

Project Outcome

Despite having a much more limited feature-set than we had originally hoped, we believe our project is an effective demonstration of what a powerful tool the InterLACE Worksheet could be. And since we believe that was the most important goal of the project, we see it as a success.

We did end up having to change a lot about our project along the way. The image analysis in particular ending up being quite different from our original vision simply because it ended up being such a huge and open-ended topic. We were disappointed not to end up being able to utilize much of the text in the wild research that we collected, but in the end we were happy to leave with a much deeper understanding of how the problem of text recognition is approached and effectively implemented despite using Microsoft’s text recognition API in our final product. The leap in difficulty from recognizing written digits to recognizing written characters was bigger than we expected, so our project scope had to be honed down to recognizing numbers and recognizing words when we have a dictionary of their possible values. We can recognize words without a dictionary now, however it is not accurate enough to be an effective demonstration.

When beginning our project, we created levels of achievement in respect to the different features we would implement for the InterLACE Worksheet. We considered Level 1 features things we would be disappointed not to have implemented by the end of the project, Level 2 features stretch goals, and Level 3 features were almost certainly not going to be implemented but functioned more to show the exciting possibilities of the project in the future. These levels can be seen below:

- Level 1
    - Locate and identify handwritten numbers
    - Locate and identify handwritten words
    - Database for class lists
    - Web app (visualization)
    - Connect with worksheet creation software created by grad student(s)
- Level 2
    - Tagging
        - Recognize student names and custom tags
    - Classroom collaboration visualization
        - Map of student collaboration for worksheet
    - Quantity detection
        - Detect how much has been written
    - Simple data comparison
        - Aggregate table, graph, heat map
    - Iterative analysis
        - Allow users to update worksheet and scan multiple times to track changes/updates (in conjunction with tagging)
- Level 3
    - Personal handwriting machine learning
        - Some sort of supervised/reinforcement learning to help analyze each student's handwriting (neural network)
            - *It may be a challenge to get reinforcement since that requires know what they wanted to write, and that requires extra interactions that may not be possible
    - Control EV3 with handwritten instructions
        - Automatically program EV3 based on students list of commands
    - Check for spelling (and math) errors
        - Provide some sort of simple feedback

We ended up implementing most of the Level 1 features. We did not implement recognition of handwritten words to the degree that we had hoped, and did not end up being able to connect our work with the current work at the CEEO. However, we do believe we will be leaving a product that can be connected to with relative ease should the CEEO use our work in further development of the InterLACE Worksheet. We were able to implement a few of the Level 2 features, including a graph of student collaboration and a graph comparing students' numerical answers in a section of a worksheet. And, as expected, we were not able to implement any of the Level 3 features. So, although we were not able to implement all of our baseline features, we reached enough of our stretch ones that we believe the project can be considered a success in respect to our original levels of achievement. To put this quantitatively, we implemented ⅘ of our baseline goals and ⅖ of our stretch goals.

Future Work

If we were to continue working on the project, our work would be relatively clear. First, we would address the existing bugs and issues in our implementation. These include:

- OpenCV throws an error when recognizing a number from an image that is wider than it is tall. This is an existing bug in OpenCV.
- Microsoft API queries take a long time, need to substitute a custom text recognition solution
- When resizing image to the size required for input images to the convolutional neural network, the image becomes skewed and leads to incorrect outputs from the CNN
- If the text in the names section or collaboration section is recognized incorrectly, a collaboration graph will not get made and the one last viewed will be displayed instead
- If you view a collaboration graph and go to a different section in the same worksheet the collaboration graph will stay on the web page instead of hiding

After fixing those, some of our immediate goals would be creating clear and concise documentation of our API so that our project could be connected to (currently we only have example API calls), separating the image processing and web application modules so that they could easily be used on their own if the CEEO only wanted to use one of them in future work, and using our own text recognition solution instead of the Microsoft API to speed up the application and increase the accuracy.

Submitted Components

Code: The repository can be found at https://github.com/johnspiva/worksheet-interlace

Poster: Our poster is currently on display at the CEEO and a PDF of the submission is included as "InterLACE Worksheet Poster"

Slideshow: Our sponsor enjoyed the slideshow that we created and asked to use it as a method of pitching the project to schools / teachers. This is called "InterLACE Worksheet Slides".

Documentation: There was extensive research that went into preparing the image processing module and web interface. All of that documentation was submitted so that future contributors can effectively pick up where we left off. This was probably the most important part of our project. A lot of time was spent better understanding the problem and learning lessons

Lessons Learned

Our assessment of the risks and potential failures was accurate. We organized our original goals into different levels of achievement, where the team had certain milestones and challenges we hoped to overcome. We found there were definitely above and beyond features we simply did not have time to implement, as well as certain hopes that did not get accomplished.

Some unexpected challenges we faced were in handwriting analysis, existing handwriting analysis did not fully cover what we were trying to accomplish, and text in the wild was too broad of a concept to fully implement for our project. Our solution to this problem was to create a hybrid of both models that accurately identified the text on the worksheet.

Another unexpected challenge was finding endpoints to connect different aspects of the project. Having different expectations and imperfect communication caused confusion in the inputs and outputs of each team member's individual components.

Time was a large factor in getting everything done. Because we are all students with other classes and other coursework, spending the amount of time required to have a perfect product by the deadline was near impossible. Having weekly meetings was very helpful to stay on track, but the team found itself scrambling at times to get everything done by the Friday meeting to be a challenge.

The class itself could be improved through starting implementation of the projects in fall semester. This would alleviate the time crunch, and it felt as though there was a lot more free time in project expectations, versus second semester there were much more hard deadlines and milestones to accomplish.

Some advice to the next year's class is to have clear communication with your group, be passionate about your project, set weekly goals, and don't sweat the small stuff. This project was rewarding and exciting to work on so make sure you enjoy what you're doing.

References

[1]Neves, Renata F. P., Cleber Zanchettin, and Alberto N. G. Lopes Filho. "An Efficient Way of Combining SVMs for Handwritten Digit Recognition." *SpringerLink*. Springer, Berlin, Heidelberg, 11 Sept. 2012. Web.

[2]Reinhard Klette (3). "Feature Detection and Tracking." *Springer*. Springer London, 01 Jan. 1970. Web.

[3]Azizi, Nabiha, Nadir Farah, and Abdel Ennaji. "New Dynamic Classifiers Selection Approach for Handwritten Recognition." *SpringerLink*. Springer, Berlin, Heidelberg, 11 Sept. 2012. Web.

[4]Bezerra, Byron Leite Dantas, Cleber Zanchettin, and Vinícius Braga De Andrade. "A MDRNN-SVM Hybrid Model for Cursive Offline Handwriting Recognition." *SpringerLink*. Springer, Berlin, Heidelberg, 11 Sept. 2012. Web.

[5]Jaderberg, Max, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. "Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition." *[1406.2227] Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition*. N.p., 09 Dec. 2014. Web.

[6]Tu, Ming, Visar Berisha, Yu Cao, and Jae-sun Seo. "Reducing the Model Order of Deep Neural Networks Using Information Theory." *[1605.04859] Reducing the Model Order of Deep Neural Networks Using Information Theory*. N.p., 16 May 2016. Web.

[7]Jaderberg, Max, Andrea Vedaldi, and Andrew Zisserman. "Deep Features for Text Spotting."*Computer Vision – ECCV 2014 Lecture Notes in Computer Science* (2014): 512-28. Web.

[8]"Text Detection With Convolutional Neural Networks." *Proceedings of the Third International Conference on Computer Vision Theory and Applications* (2008): n. pag. Web.

"Stable Marriage Problem." *Wikipedia*. Wikimedia Foundation, 27 Apr. 2017. Web.

Max Jaderberg , Karen Simonyan , Andrea Vedaldi , Andrew Zisserman, *Reading Text in the Wild with Convolutional Neural Networks*, International Journal of Computer Vision, v.116 n.1, p.1-20, January 2016 [doi>10.1007/s11263-015-0823-z]

Falk TH, Tam C, Schwellnus H, Chau T. Grip Force Variability and Its Effects on Children's Handwriting Legibility, Form, and Strokes. ASME. *J Biomech Eng.* 2010; 132(11):114504-114504-5. doi:10.1115/1.4002611.

Hynes, Dr. Morgan M., Dr. Ethan E. Danahy, and Ms. Danielle Dowling. *AC 2012-4150: The InterLACE Project: Examining the Barriers to Implementing Collaborative, Inquiry-Based investigations* (n.d.): n. pag. Web.

Beatty, I.D., et al., Designing Effective Questions for Classroom Response System Teaching. American Journal of Physics, 2006. 74(1): p. 11.

Finn, Nicola. "Education Technology Hardware Spend in K-12 Increases 7% in 2015". *Futuresource Consulting*. 14 April 2016. n. pag. Web.

DeWitt, P. (2013). Are schools prepared to let students BYOD? Education Week.

Cytowic, R.E. (2015). Your brain on screens. . The American Interest, 10(6), 53-61