NETS 213 Project - ESP Image Reveal Game

Oliver Ferry, Juliette Hooper, Yoonduk Kim, Joel Lee

[logo]

Description: ESP Image Reveal Game is a game with a purpose for identifying informative regions of images in object detection tasks

Project Objective and Review of Similar Projects

Our goal is to create a dataset of informative image regions for object detection. Pixels of an image vary in their importance for identification. Certain areas may contain features that are helpful for detection (eyes of a cat), while others are only useful in conjunction with others (jockey on a horse) or convey no information at all (background).

The question of which features of an image hold importance for detection has been a popular subject of research in computer vision. Zeiler & Fergus (2014) visualized feature activations of AlexNet, to demonstrate that hidden layers in the network learn to identify generalizable features of images that are robust to transformations. Simonyan et al. (2014) proposed a technique for generating saliency maps for image classes and Zintgraf et al. (2016) presented a method for displaying the parts of the input image relevant to classification output. Meanwhile, many studies have been conducted on how humans process images in the field of psychology and neuroscience. Extensive literature of interviews, image description tasks, and eye tracking have advanced our understanding of how humans recognize one object from another.

In this project, we aim to bridge the gap between the two fields through a game in which crowdsourced human participants mark regions of images that they deem are most informative. Relatively little work has been done on comparing the object detection mechanisms of humans and deep neural networks. For instance, it is unknown whether human judges agree with neural networks on the importance of certain visual features. In this project, we develop a game with a purpose to collect regions of images that people find most useful for object identification. The first dimension denotes which area of images are perceived to be informative by the sender of information. The second dimension is a performance measure of how well the recipient identifies the object in the image given the information from the sender.

Project Overview

We generate our dataset through a two-step game with a purpose. The game is divided into two crowd-sourced components. In the first game (Annotation), a player is presented with two versions of the same image, one blurred and one unblurred. Each click on the blurred image reveals a small area around the cursor. The player's goal is to make a series of clicks such that another player can guess the object in the image as quickly as possible. The sequence of clicks on each blurred 'block' of the images are tracked for aggregation and analysis.

The second game (Guessing) involves another player, who is given only the blurred image and an input box for submitting guesses. After each guess, a new area of the image is revealed in the order of clicks made by the player in the first task. The player of the second game must correctly identify the object in the blurred image in as few guesses as possible. The list of guesses are processed to calculate informativeness of each corresponding reveal.

We ran a study in which 100 responses are collected in each game for 5 input images. The images are sourced from the Common Objects in Context (COCO) dataset. While it is possible to rely on human judgment for quality control and aggregation for this particular study (~1,000 responses), we developed automated quality control methods to allow for future scalability. Quality control for the annotation game is done by utilizing the object bounding box feature of the COCO dataset. We use this information to assign scores for the player's accuracy. While clicking on different parts of the object may depend on personal preference, clicking outside the object can be considered to be false signals. Quality control for the guessing game is more straightforward. The number of images the player guesses correctly over the course of their HIT may be used to judge the quality of the response, though we note that the performance may be affected by the quality of 'input click sequences' from the annotation game.

Aggregation is likewise handled automatically via Python's boto3 for managing MTurk HITs, pandas for data cleaning, and pyplot for visualization. We developed the two games from scratch using React.js and Gatsby.js framework. The game is hosted on netlify and can be played at:

https://image-reveal-game.netlify.app/

A detailed flowchart of the architecture is shown in Appendix A.

Type of Project and Focus of our Effort

For the purposes of this project, we set up a framework for collecting data on image feature informativeness through a fun game, inspired by Luis von Ahn's Game with a Purpose (Von Ahn, 2006). While the current version is designed for paid workers on Amazon Mechanical Turk platform, we plan to improve on this work to make it available for any participants to play online for pure amusement.

This project may be considered both a human computation algorithm and a social science experiment with the crowd. The aggregated annotations on each image provide insights into what features of an image people believe are most informative to the receiver. Analyzing the performance of the guessers can demonstrate which pixels have the most impact on identification. The perceived importance score and the guessing performance score both may serve as useful data for future research on comparing the human and computational models for object recognition.

Additionally, we will conduct several exploratory analysis on what visual features are perceived to be helpful, based on the result of our pilot study. From 200 participants, we collected 500 click sequences and 610 guess sequences for 5 input images. We find that for animals, the faces are considered to be the most informative features. On the other hand, objects such as bicycle and pizza have a wider variance of clicks. Pizza had the most abstract and dispersed annotations due to its uniform shape.

In the future, we plan to expand our input dataset to comprise the entirety of the images COCO dataset. We expect our data to have a wide array of possible applications. One would be to use the human-informativeness measures to finetune object detection algorithms. Another use would be for developing accessible HCI. For people who are visually impaired, the data can be utilized for zooming in to important parts of images. For internet users in countries with limited network access, a new video encoding scheme may be developed to vary the resolution depending on their informative value.

Final Presentation Video: https://vimeo.com/547697726

Github Repository: https://github.com/kimyoonduk/image-reveal-game

The Crowd

The intended members of the crowd for our project are online players who are looking for an image labeling game to play. In our experiment, we ran the game with Amazon Mechanical Turk workers. The users that completed our image annotations were real. However, our crowd was simulated in the sense that they were paid to complete tasks rather than being organically attracted to our platform. Since our crowd was simulated, the collection of data was slightly different from the case of a real crowd. In the case of a real crowd, we would be hosting the image labelling platform ourselves and users would access our platform via the web. In our simulated crowd, the game tasks were embedded into Mechanical Turks HITs where workers would complete the tasks. Additionally, for a real crowd we would need to rework the home page and labelling systems to convey more clearly to users that they are playing a game. This could include a leaderboard for high scores and an improved user interface.

Incentives

Throughout our experiments, we gathered a total of 200 unique participants. Task 1 accounted for 99 of the participants while task 2 accounted for 101 participants. Since these workers were obtained via Mechanical Turk, their primary motivation was monetary. Although, factors such as enjoyment and altruism may have contributed as well. To address the issues of underpaid workers on Mechanical Turk (Fort et al., 2011), we set up our rewards to match the US Federal Minimum Wage of $7.25 an hour (Department of Labor, 2021). A pretest on the students of NETS 213 class showed that each task took 1.5 to 2.5 minutes. We paid each worker between $0.18 and $0.36. A preliminary analysis of our output showed that the amount of rewards did not affect worker performance.

In order to incentivise a real crowd rather than a crowd obtained over Mechanical Turks, we have taken the approach of gamifying the data collection so as to incentivise the crowd through personal enjoyment. In order to make the game enjoyable enough to generate traction with online users, there are several facets of our application that need to be bolstered.

First would be the overall UI of the website. We would need to make the website more visually appealing and arcade like so as to draw in users off the web. Next, we need to give players a sense of score and competition to incentivise them to continue playing. This can be achieved through an online multiplayer aspect to the game where players compete against each other for the highest score on the leaderboard. We could also have a fake multiplayer aspect to the game which many .io games have been using. In this way players have the illusion that they are playing with another online player in real-time when in reality they are asynchronously playing with another player.

Further analysis does need to be conducted on the efficacy of gamifying the data collection. The main hurdle of this approach is being able to garner a large enough crowd to collect data. The game may not be perceived as enjoyable enough to gain traction among users. In the future, we aim to provide an open platform where anyone can volunteer to play the game. This will allow us to gauge how many people are incentivized to play for entertainment. Updates to the game's achievement design or visual effects may be required.

What the crowd gives you

The crowd will annotate the image data by selecting the pixels of an image with the highest importance. The crowd will also provide quality control on the image annotation data by attempting to guess the content of an image using only the annotated pixels. It is possible that a machine learning model could annotate the images. However, in order to generate this model we would need a large dataset of annotated images.

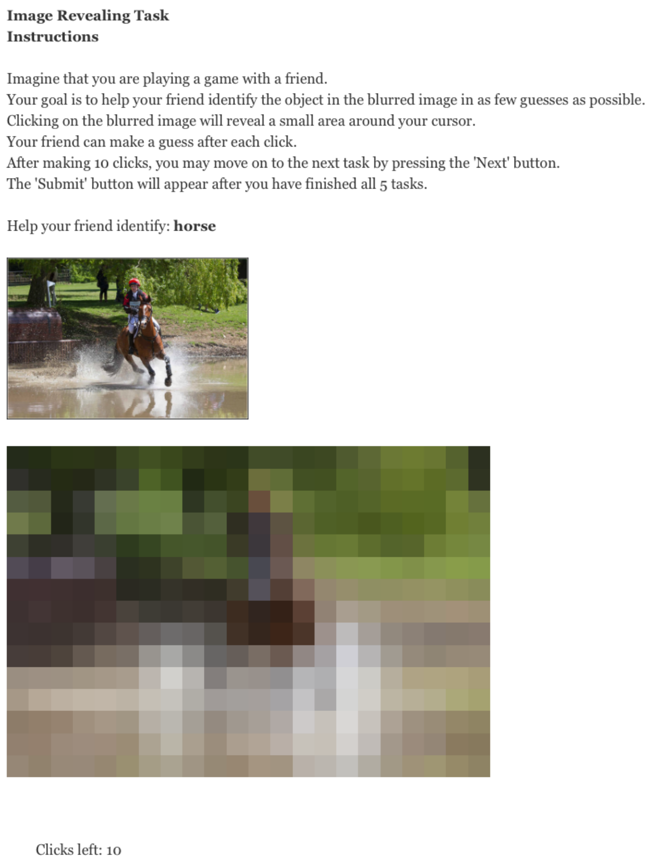


Image Revealing Task
Instructions

Imagine that you are playing a game with a friend.
Your goal is to help your friend identify the object in the blurred image in as few guesses as possible.
Clicking on the blurred image will reveal a small area around your cursor.
Your friend can make a guess after each click.
After making 10 clicks, you may move on to the next task by pressing the 'Next' button.
The 'Submit' button will appear after you have finished all 5 tasks.

Help your friend identify: **horse**

Clicks left: 10

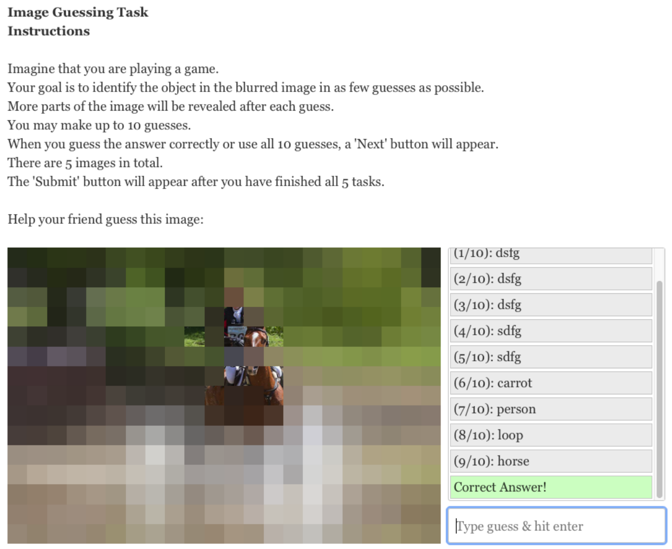


Image Guessing Task
Instructions

Imagine that you are playing a game.
Your goal is to identify the object in the blurred image in as few guesses as possible.
More parts of the image will be revealed after each guess.
You may make up to 10 guesses.
When you guess the answer correctly or use all 10 guesses, a 'Next' button will appear.
There are 5 images in total.
The 'Submit' button will appear after you have finished all 5 tasks.

Help your friend guess this image:

(1/10): dsfg
(2/10): dsfg
(3/10): dsfg
(4/10): sdfg
(5/10): sdfg
(6/10): carrot
(7/10): person
(8/10): loop
(9/10): horse
Correct Answer!

Depicted above, we have our user interface for the image guessing and image revealing task. In the Image reveal task, we pixelate the input image and then prompt the participant to click on pixels that are most important in conveying what the image depicts. When the participant clicks the pixel, the pixel will become unpixelated revealing to the participant the original image in that area. The participant has a total of 10 pixels to reveal. In the image guessing task, a participant has a dialog input box where they may make up to 10 guesses on what the image contains. When the participant guesses incorrectly, an additional pixel is unpixelated to the participant and they may continue making their guesses.

Skills

Our tasks do not require the crowd workers to have specialized skills. For Task 1, all they need to know is to have a basic understanding of common English nouns and what they represent. For example, to know what a bicycle or a pizza means and to be able to identify it in a photo. For Task 2, the skills are mostly the same because they are simply identifying the deblurred image. The skills of individual workers did not vary significantly because the task performance was mainly reliant on the understanding of the English language. Towards that end, we did not end up analyzing the skills of the crowd, by some pre-test or authentication as other papers have mentioned. Because of the simplicity of the tasks as stated above, we simply had them start the task with no check for skills.

Quality Control and Aggregation

In general, the quality of what the crowd gives us is of a moderate level of concern. Given that our tasks are fairly simple, we wanted to create equally as simple ways to calculate the quality of our workers' inputs. This means we don't use a gold standard or any complex Quality Control strategies. For Task 1, we check to see how many of their deblurring clicks are within the bounding box of the subject, and for Task 2, we check to see out of the five images how many of them they can correctly guess.

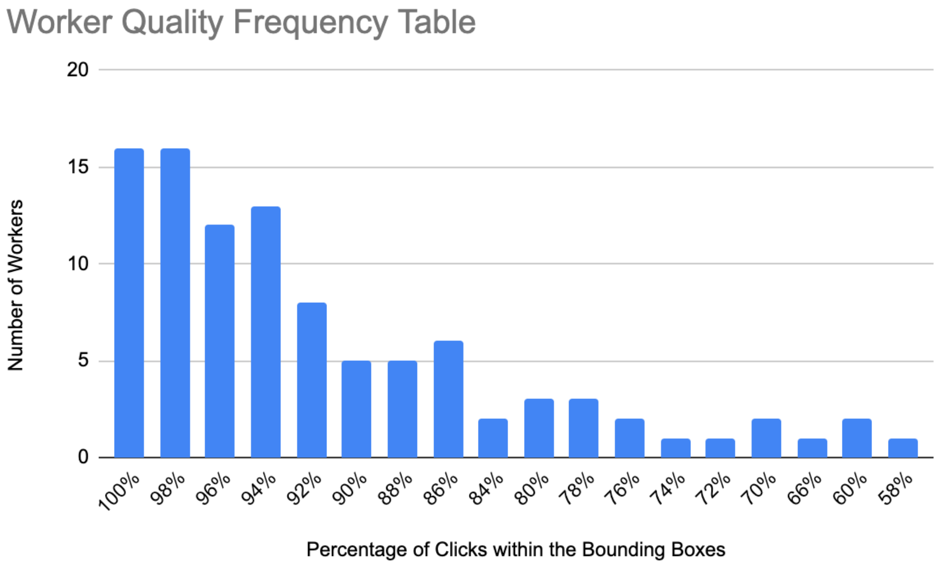
Task 1 Quality Control

For the quality control in our first task, we wanted to check and see how often workers were selecting boxes in the area of the actual subject of the image. In the COCO dataset, there is a field in their metadata.json for their objects called "bbox" or bounding box. This is the approximate rectangle coordinates for each of the objects in the photo. An original iteration of Task 1 had the collection of clicks being done as coordinate points that would be recorded. Here, the quality would be checked by creating a Shapely polygon with the bbox coordinates and a

Shapely point with the coordinates of the user click, and then using the Polygon.contains() function to assess whether or not the user clicks were within the polygon.

However, the iteration of data collection we ended up using was one where each blurred box was a coordinate point which we created, anywhere from 1 to around 12-17 based on the height and width of the image; a 2D array of blurred boxes. Then, a user click would input as a coordinate point (ex: [4,12]) for which blurred box they decided to deblur. Here, we decided to check the quality by looking at the bounding box given by the COCO dataset, and approximating the bottom left and top right coordinates when translated to our blurred box 2D array. With these two points (ex: [[4,2],[6,10]]) and our user clicks, we could use a simple geometric algorithm to figure out whether or not the user click was within the “bounding box” that we had translated from the COCO dataset. We understand that if scaled up and automated, we would want a way to translate effectively COCO’s bounding box coordinates to our blurred coordinates without human interaction, but for the case of our five images, we thought it would be best to do this part manually.

Each user has 10 clicks for each of the five images, so we were able to calculate how many clicks were within our bounding box divided by their number of total clicks. From there we were able to calculate the quality of each worker. This was similar to the way we calculated quality in Homework 7. Overall, the quality of our workers for Task 1 was very good. The workers had an average quality of 0.909 and a median of 0.94. 16 of our 100 workers had 1.0 quality. We only had one score below 0.6 in worker quality, and so we decided to keep all of the user clicks for the guessing phase in Task 2. We put in this quality control module because we wanted to catch if users were making wild clicks or not trying to deblur the image at all. If we had one that was extremely low, such as less than 0.25, then we would’ve considered removing it under that basis, but the quality range given by the workers seemed satisfactory for Task 2.

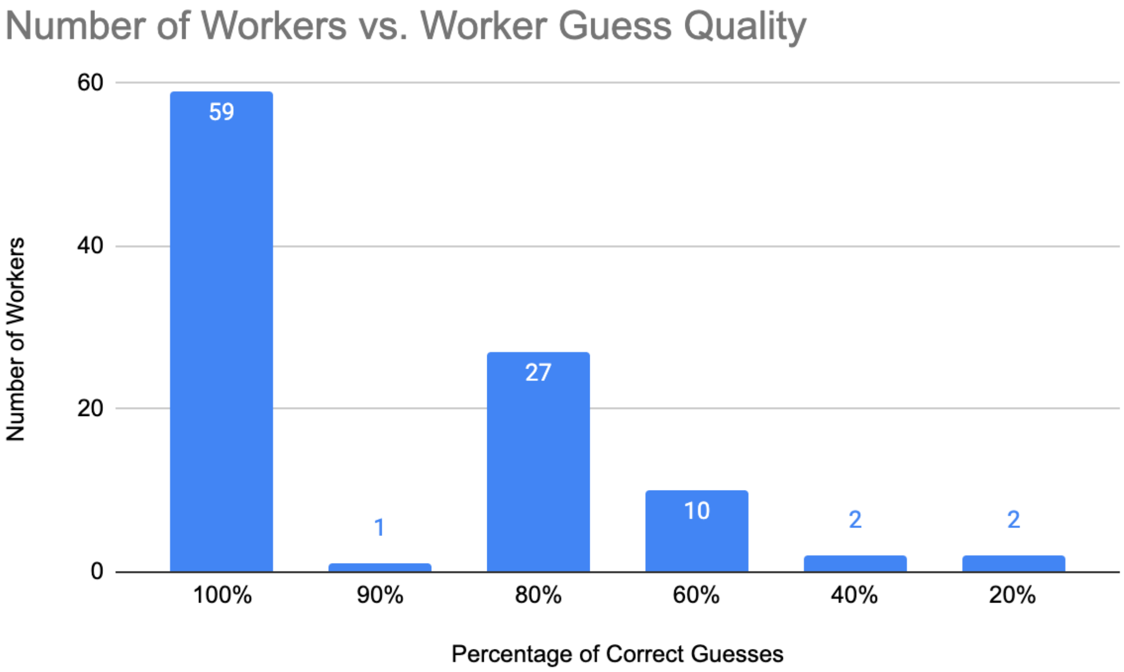


Task 1 Aggregation

The actual aggregation in Task 1 in order to feed the clicks into Task 2 was fairly simple. As an output to MTurk from Task 1, we were given a list of 10 coordinates that represented the 10 blurred boxes that the users deblurred. This in turn was fed into Task 2 where we were able to use these lists to create a sequential ordering of deblurred squares for users to guess the label of the blurred image. However, something we additionally wanted to consider was aggregating the data only from Task 1, in order to see what all of the workers were deblurring for each image, regardless of any guessing that would happen afterwards. Towards this end, we also used a notebook called “task1-aggregation” in order to visualize a heatmap of each of the coordinate clicks of every user on top of the image. We analyze this data in the Project Analysis section.

Task 2 Quality Control

The Quality Control module for Task 2 consisted of a simple check to see how many out of the five blurred images presented were the users able to accurately identify. Because we already had the answers we were able to check whether or not any of their maximum of 10 guesses were the label that we were looking for (“cat”, “pizza”, “bicycle”, “horse”, “giraffe”). If none were the correct label, then they got that image task incorrect. Then this was calculated into a percentage. The results showed that a large majority of the users were able to get the label correctly from the blurred images. Only 4 users did not get over half of the labels correct, and so we removed those guesses from the final data aggregation. Of course, it can be the case that the deblurring was poor enough that the users could not guess the object, but we quality checked the users in Task 1 and also the users in Task 2 are given five different deblurring sequences to be tested on.



Task 2 Aggregation

Task 2's aggregation again involved a heatmap, much like Task 1. However, in this case, the metric we wanted to use for the intensity of the heatmap was which specific blurred box becoming unblurred led to the correct guess of the user. Task 2's output on MTurk was formatted as a list of string guesses of max length 10 (one guess for every deblur of a Task 1 worker). Additionally, in every row of the HIT, we had the Id of which deblurring sequence was being used in a HIT in Task 1. Thus, we can line up each guess with each sequential deblur and figure out which deblurred coordinate box led to the correct guess. Thus every "winning deblur" was incremented by 1 in the intensity for the heatmap, because that specific box that was deblurred led to the user being able to guess the label correctly. We were able to visually see this heatmap overlaid on top of the image with the help of matplotlib and its heatmap function. We created a numpy 2D array that represented the blurred boxes from Task 2. And for each "winning deblur" we incremented it in the 2D array and plotted it over the original image. This graphical analysis is done in the Project Analysis section.

Scaling Up

The problem of obtaining a distribution of informativeness for each image would immensely benefit from having a large dataset. Annotations for multiple variations of the same object makes it possible to identify its important features depending on transformations in perspective, lighting, and other visual variations. For instance, a cat when seen from below may have different informativeness ranking of features than when seen from the side, while the logic would not apply to a spherical object such as a soccer ball.

This pilot study is conducted on 5 input images of varying objects - cat, giraffe, horse, bicycle, pizza. We have collected 100 click and guess sequences per image. From this study, we’ve found that 100 is a sufficient number to meaningfully identify the pixel regions people deem important for object recognition, though somewhat lacking in detail for more abstract objects (pizza). In the near future, we hope to create a baseline dataset for all 80 objects in the COCO Dataset.

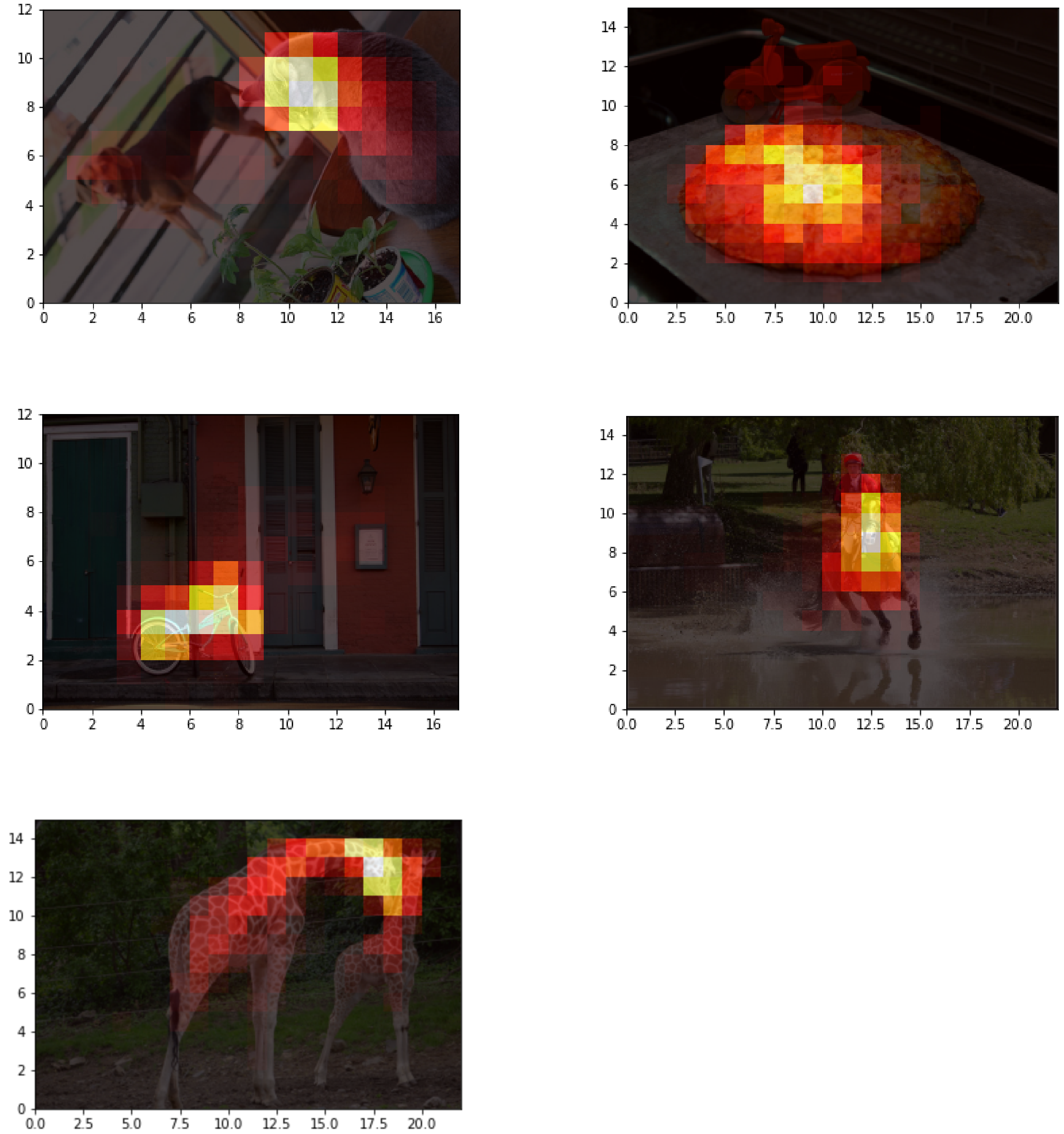
Assuming that we collect 5 variations on each object, the total number of tasks would amount to 40,000 (100 * 80 * 5) annotation labels and 40,000 guess sequences. We paid each MTurk worker between $0.18 to $0.36 per 5 images. Adding the 40% fees to the platform, this means that each image label costs around $0.05 - 0.10 (0.36 * 1.4 / 5). Therefore, in order to pay crowdworkers to generate sufficient baseline data for all 80 objects in the COCO dataset, we estimate that $4,000 - 8,000 would need to be spent (0.10 * 80,000).

The baseline dataset of 80,000 annotations would still be a drop in the bucket considering the body of available data on COCO. According to the documentation, COCO provides 1.5 million object instances over 200,000 labeled images. This means that each object has an average of 18,750 variations represented on the dataset, while our baseline only has 5. Increasing the number of image representations for each object will enrich the data for generalized recognition of key features. Collecting click and guess sequences for all 1.5 million instances via MTurk would cost approximately $300 million (1.5 mil * 100 * 2), which is extremely costly. The operational cost of handling traffic to the game and managing the database would add to the cost.

Fortunately, we believe that the purpose of the dataset and the design of the collection process opens up doors for voluntary participation. First, the goal of this dataset is to enrich people’s lives by improving existing image recognition algorithms and assist in efforts to create accessible HCI for people that encounter trouble processing images. This would encourage people to participate based on altruistic motivations. In addition, the game has a large room for improvement in the entertainment aspect. Setting up high scores, adding visual effects, and social functions may attract people wanting to have fun. Our ultimate goal is to create a platform in which the game itself is rewarding by itself, even without monetary incentives.

Project Analysis

Task 1 Analysis

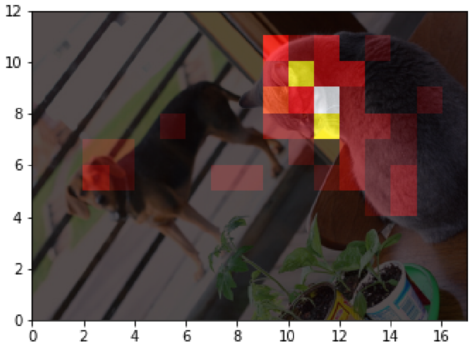


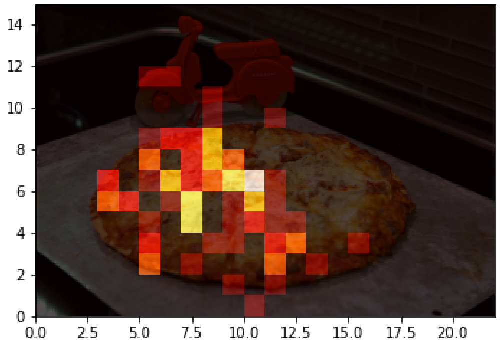
(left to right, top to bottom: heatmaps of cat, pizza, bike, horse, giraffe. Intensity is calculated as an aggregation of all the clicks of coordinate squares from users in Task 1.)

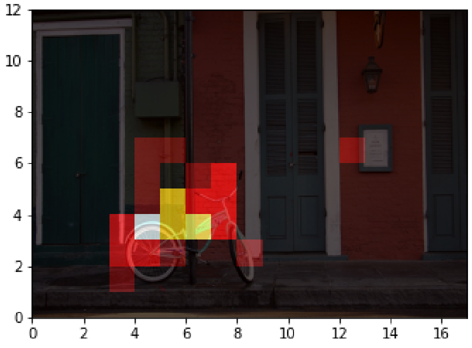
In Task 1, we wanted to aggregate the results of all the clicks the users made in order to deblur the image for the users in Task 2 to guess the label. In this way, we don't actually need the Task 2 results in order to display this heatmap. We think that this part of the project worked pretty well and we know this because of the accurate heatmap that is produced by the results. As we can see with all five pictures, the heatmaps are generally along the boundaries of the subject of the image in question. The heatmap does well to display the hierarchy of importance for each of the images. This is represented best with the giraffe photo, as the white and yellow boxes are centered around the head of the giraffe, signifying a high intensity, and the intensity is lowered as you move down the giraffe's neck and body. For all three of our animals, the general head and face area of the animal is what is most intense, and that intensity decreases as you move away from the head and towards the body.

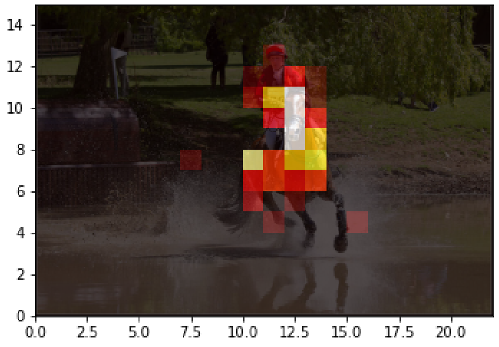
For the pizza and the bike, the results are a bit more abstract. The bicycle seems to have the most intense squares near the body of the bicycle, the top of the back wheel, and the handlebars. The bottom half of the wheels are noticeably less intense. Perhaps this means that the upper half of the bicycle is most distinguishable, or that not all of the wheels need to be revealed to give a user the message that it is a bicycle. For the pizza, there is a clear bias towards the center of the pizza, and from there the intensity seems to move outward mostly to the left of the pizza. Although the ordering isn't displayed in any manner in the visualization, it looks as though the users would start in the center of the pizza, and then move outwards to include the crust of the pizza. However the lack of intensity on the right side of the pizza isn't quite discernable or understandable to us in any significant way.

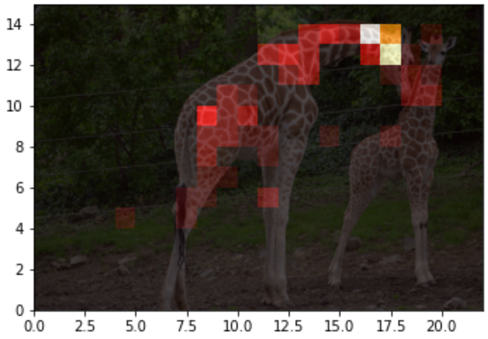
Task 2 Analysis











(left to right, top to bottom: heatmaps of cat, pizza, bike, horse, giraffe. Intensity is calculated as an aggregation of all the blurred coordinate squares that was revealed at the same time as the winning guess of the user in Task 2.)

In Task 2, we calculated intensity by assigning a score to each of the squares being deblurred based on how many times that square was the “winning square”; meaning that that square being deblurred led to the user in Task 2 correctly guessing the label. Thus, as we stated in the aggregation section for Task 2, we were able to line up a user’s correct guess with the exact square that was deblurred. We did this aggregation over all of the guesses and created the same type of heatmap as we did for Task 1.

The results for this task were more precise than the results in Task 1, which were generally broader and expansive. For the cat, the squares that seemed to have prompted the correct guesses were the bottom half of the cat's face, including its whiskers and its mouth. What’s crucial to note for this analysis however is that we don’t particularly pay attention to the possible previous squares that have led to this “winning square”; that is, a particularly intensely colored square could have led to a correct guess because it was revealed along with any number of previous squares already being revealed. We don’t take into account the order of which the winning square appears, whether it be the first (meaning the only square needed to guess the label correctly) or the last (the tenth and final square that prompted the user’s correct response). It’s important to note that still the most intense squares are on the head of the cat, but that the number of squares with higher intensity are lower.

The giraffe is another example of this, where the most intense squares are not directly on the face of the giraffe, but lie generally around the head area of the giraffe. One hypothesis is that many of the clicks first reveal the eyes or facial features, but more context is needed around the head for users to guess confidently what animal it is. This line of thought would line up with the giraffe and the cat images, as the distinctive facial features (such as the eyes) aren’t the most intense squares, but rather the most intense squares are around the head. The horse image seems to do well in this type of aggregation however. Distinctively the most intense squares are the three squares which directly highlight the face of the horse.

As for the pizza and the bicycle, the results are still a bit abstract, but noticeably different from the results in Task 1. The pizza seems to be more scattered, specifically around the edges of the pizza. Again our conjecture of order might appeal here: Maybe the middle of the pizza was first deblurred, but the crust and the edges of the pizza were needed to fully convince some of our users that it was in fact a pizza. The bicycle also serves as an interesting example because of its difference from Task 1. In Task 1, the bicycle handles were clicked often and thus intensely highlighted. However in Task 2, the handles are noticeably less intense than the top part of the back wheel and the body of the bicycle. This shows us that although many users click on the handlebars of the bicycle in order to make it identifiable to others, it is actually the wheel and the body of the bicycle that help users most in making the correct guess.

Overall, there weren't many major changes between what we originally proposed and our final project. However, it's clear that after this analysis, there are many more ways in which we can imagine and visualize this data. Something evident that we would want to further look at is considering order in the metric of intensity. If the square revealing the top part of a bicycle is the first square deblurred and it is guessed with only that square deblurred, then that should have a higher intensity than if it is the tenth square deblurred, and the user guesses correctly with that square along with the previous nine squares that have been deblurred. Our Task 2 aggregation currently does not differentiate between these two, and that is a limitation of our product in its current form. However it's not entirely clear how to best go about it, and also whether or not to give, in this case, the previous nine squares also some points in intensity. Figuring out the details of exactly how to aggregate the data in a heatmap was perhaps the biggest challenge of the analysis part of the project, and there are certainly many more ways to do it.

Technical Challenges

The project's user interface required a substantial technical component. Each task was unique in the sense that no Mechanical Turks template could provide the correct interface to collect the data that we were looking for. We needed to use the framework Gatsby.js such that we could write the logic behind image revealing in React and statically generate it to embed inside the Mechanical Turks HITs.

One of the major technical challenges was that each image from the COCO dataset was of different aspect ratio and resolution. As a result we needed a way to consistently pixelate the image and then reveal it on click on the frontend application. Our solution was to write a python script that would preprocess each image generating a 2D array of RGB values that would be passed on to the client which would then be able to use the 2D array to cover the image with the right number of colored pixels. On the Quality Control side of the project we also faced technical challenges regarding using bounding boxes provided by the COCO dataset to validate the crowd data as well as writing scripts to generate heat maps on the provided datasets.

References

Department of Labor. (2021). https://www.dol.gov/agencies/whd/minimum-wage/state

Fort, K., Adda, G., & Cohen, K. B. (2011). Amazon mechanical turk: Gold mine or coal mine?. *Computational Linguistics*, *37*(2), 413-420.

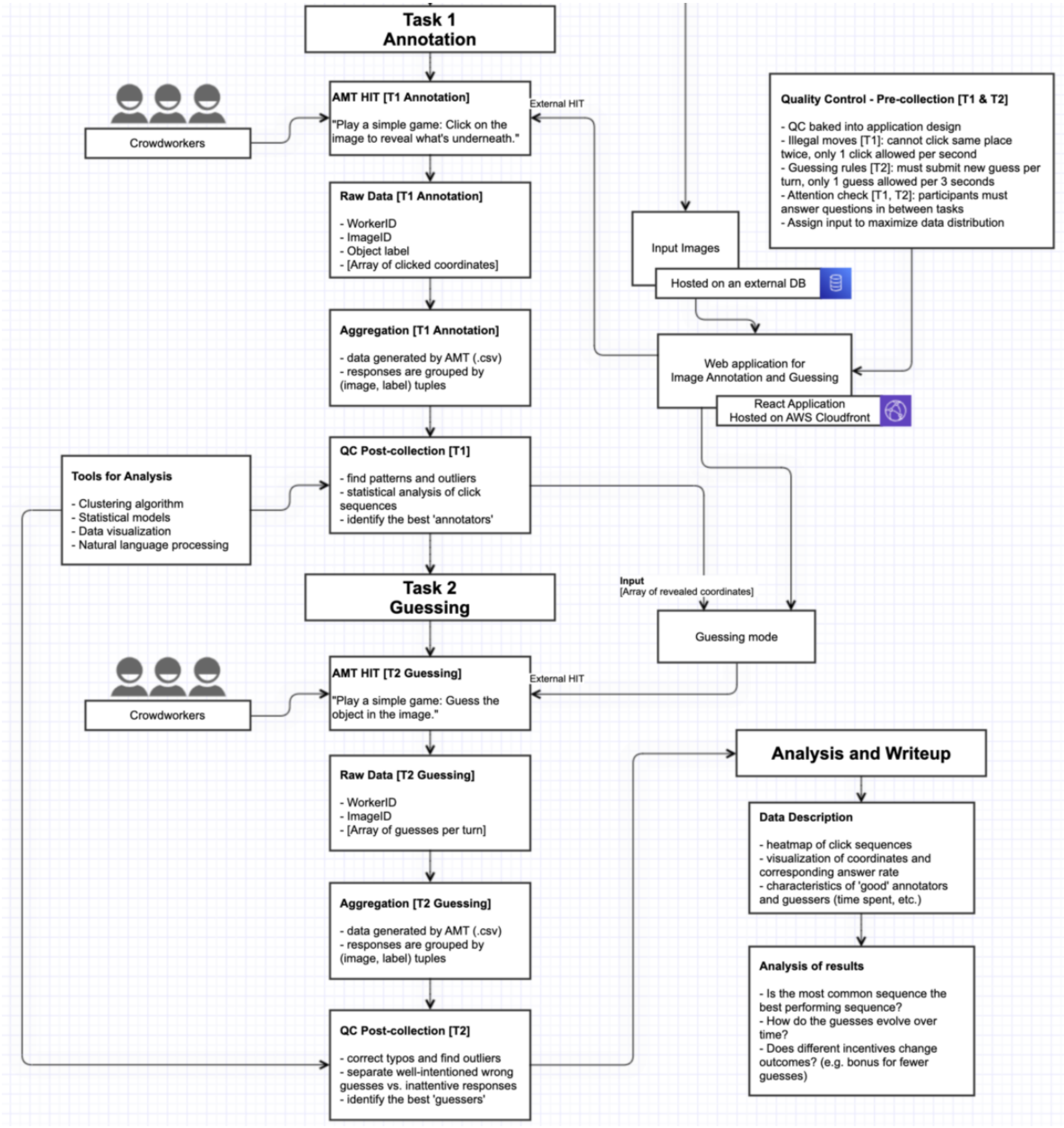
Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps.

Von Ahn, L. (2006). Games with a purpose. Computer, 39(6), 92-94.

Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer, Cham.

Zintgraf, L. M., Cohen, T. S., & Welling, M. (2016). A new method to visualize deep neural networks. arXiv preprint arXiv:1603.02518.

Appendix A - Project Flowchart



AMT HIT [T1 Annotation]

"Play a simple game: Click on the image to reveal what's underneath."

Raw Data [T1 Annotation]

- WorkerID
- ImageID
- Object label
- [Array of clicked coordinates]

Aggregation [T1 Annotation]

- data generated by AMT (.csv)
- responses are grouped by (image, label) tuples

QC Post-collection [T1]

- find patterns and outliers
- statistical analysis of click sequences
- identify the best 'annotators'

Quality Control - Pre-collection [T1 & T2]

- QC baked into application design
- Illegal moves [T1]: cannot click same place twice, only 1 click allowed per second
- Guessing rules [T2]: must submit new guess per turn, only 1 guess allowed per 3 seconds
- Attention check [T1, T2]: participants must answer questions in between tasks
- Assign input to maximize data distribution

Tools for Analysis

- Clustering algorithm
- Statistical models
- Data visualization
- Natural language processing

AMT HIT [T2 Guessing]

"Play a simple game: Guess the object in the image."

Raw Data [T2 Guessing]

- WorkerID
- ImageID
- [Array of guesses per turn]

Aggregation [T2 Guessing]

- data generated by AMT (.csv)
- responses are grouped by (image, label) tuples

QC Post-collection [T2]

- correct typos and find outliers
- separate well-intentioned wrong guesses vs. inattentive responses
- identify the best 'guessers'

Data Description

- heatmap of click sequences
- visualization of coordinates and corresponding answer rate
- characteristics of 'good' annotators and guessers (time spent, etc.)

Analysis of results

- Is the most common sequence the best performing sequence?
- How do the guesses evolve over time?
- Does different incentives change outcomes? (e.g. bonus for fewer guesses)