Architexa project portfolio

Contributors

Emmanuel Adegoke	Fahad Alsharaf	Matt Paver	Meshari Alshammari	Stephen Bromley	Wilf Morlidge

People and roles

The teams

For most of the project, the team was split into 3 sub-teams. Matt leads the custom dataset, Fahad and Meshari compose the website design team, and Wilf and Stephen are on the neural network team. Emmanuel initially joined the neural network and floated between teams, supporting the team with the most work to do then.

Major Tasks and Achievements

Task	Major Role	Minor Role
Generator Design	Emmanuel	Wilf, Stephen
Discriminator Design	Wilf	Emmanuel, Stephen
Training Implementation	Wilf, Stephen	
Hyper-Tuning Implementation	Stephen	Wilf
UI & UX	Meshari	Matt
Designing The Website	Fahad, Meshari	Matt
Website Login Implementation	Matt	
API Implementation	Fahad, Meshari	
Gathering The Dataset	Matt	
Dataset CASE Tool Design	Matt	
Graphic Design	Meshari	
Domain Name Purchased	Meshari	
API hosting Plan Purchased	Meshari	
Documentation primary author	Wilf	

Fahad and Meshari came into this project with moderate programming skills but little experience in web development. Through extensive research and help from Matt, a more experienced web developer, they created a fully functional website complete with a login feature and an API call. Meshari has experience in graphic design and made the Architexa logo and graphic images displayed on the website. The website was hosted on Firebase, and using their free spark plan proved to be a suitable choice for the scale of our project. However, hosting the AI model as an API was a challenge, because of the model requiring specific TensorFlow libraries, and a moderate amount of compute to run it. Furthermore, we needed a free option to work with. After researching between services, The API was developed initially as a

Node.js RESTful API using the Express framework. When testing on a local environment everything was working fine, but when deploying to a hosting provider's environment an issue occurred where the AI model file did not run. After some trouble shooting it was discovered that the hosting service could only provide a single runtime environment per hosting web service. Thus, we decided on adapting the API into the Flask framework, to import the AI model file contents directly into the Flask app file and run it. The free plan did not provide enough compute to run the AI model. Therefore, Meshari offered to upgrade the subscription plan to a higher one. This worked and when the AI model API is called from the website it generates an image and displays it.

Wilf and Emmanuel started with some knowledge of neural networks but not much beyond that taught in COMP219. They led research and development into both architectures used, with Wilf then making the first training script and the initial discriminator architecture, which provided a great starting point for the later development of the hyper-tuning (as well as assisting with research for the later stability updates); furthermore he wrote the bulk of all the documentation, presented the demonstration, chaired meetings and set weekly agendas. Emmanuel showed great adaptability, slotting into whichever team he needed.

Matt oversaw the collection of the dataset and the creation of tools to augment it. This proved to be quite a challenge due to scraping the internet for images. The solution was to use a combination of Python and .yaml files, which is an area of coding none of us had any experience with. The result was having a completely custom dataset of over 2000 high-resolution images with the tools to add classes if needed freely. The images used were generated in bulk using an image generator with Midjourney to ensure that the images were of similar training standards. We did not run into any copyright issues, as Matt owned the IP of the images.
However, it was understood that it could lead to model collapse due to training on AI-generated content. Still, it is a proof of concept where many images were required. With a constraint of time and resources, this would be the first iteration of this, where it is to show what is possible rather than spending countless days manually finding images that would be suitable for the model. Matt created an image scraper using Pinterest but noticed too much variation, leading to the alternative method Midjourney.

Stephen came into this project with expertise in this area, having previously worked on a speech-to-text model. He wrote most of the hyper-tuning code and utilized his experience to interpret the model's losses and outputs to help steer the training and tuning process. He identified that the current training process from the CGAN paper was too unstable and caused artifacts and overfitting, so he spearheaded the switch to the more robust WGAN-GP framework.

Application overview:

Application domain:

Architexa is commonly known as an AI image generator (similar to products such as Craiyon, Midjourney or Imagen). This means it takes textual prompts from a user and generates images corresponding to them.

The purpose of such systems is to allow those without artistic or technical drawing skills to create images faster and with a higher quality than would be possible for a human.

Architexa differs from other applications in this domain because our API targets architects. We hope our nascent model generates detailed architectural diagrams in relation to highly specific prompts. Also, the project contains case tools designed to address the challenges of creating such a model.

Therefore, Architexa addresses both the noted lack of CGAN models for technical drawings, the difficulty and non-deterministic nature of hyperparameter optimisation, and the huge time taken to construct proprietary datasets (especially multi-label datasets) via conventional image dredging, farming, and hunting.

Types of users

Non-technical CGAN users

This user type refers to members of the general public with no technical background who may wish to use the current version of the CGAN model to generate images of buildings as a labor-saving device when creating artwork, or for entertainment purposes. (which are the current main use cases for non-technical users of CGAN models)

Users of this type may access the model by navigating to our website's main page, submitting a simple prompt corresponding to one of our supported classes, and waiting roughly 75 seconds until they are presented with four options of generated images corresponding to the submitted prompt.

CASE tool users

This user type refers to other CGAN developers who wish to use our Distributed optimisation CASE tool to improve the performance of their models or our database extraction CASE tool to reduce the time taken to create databases for their models.

Users of this type may access our case tool by navigating to our project's GitHub repository and installing the appropriate case tool by following the instructions found in the project README files.

Database users

This user type refers again to CGAN developers, but specifically to those who may be working on similar products and may wish to bypass the database development process entirely by using our dataset. This is not unlikely, given that no other multi-label datasets of external shots of buildings are readily found to exist.

Users of this type may access the database by once again navigating to our GitHub repository and accessing the subdirectory of the database extraction CASE directory, which contains our dataset (before downloading it).

Technical CGAN users: (future development)

This user type refers to our initially targeted user type and consists of architects and other appropriate professionals who may (once the model's dataset is expanded and fuzzy logic processing is added) wish to use our model to generate technically accurate designs for buildings based upon detailed textual prompt input.

Users of this type would theoretically need to create a 2 or 3-paragraph English language description of the building they wish to design (remembering that this is still much less work than creating a blueprint) and then feed it into our website in a similar manner to our non-technical users, before waiting several minutes to be greeted by an appropriate mockup design for their building. Unfortunately, this will need further refinement, which will be explained later in this document.

Brief description of major components

Neural networks

The core of the system consists of two neural networks: A generator which upsamples noise into an image using stabilized transpositional convolution and a discriminator (sometimes referred to as a critic) which down-samples generated images into feature maps to give each image a score based on how "real" it looks.

These work in tandem to form an adversarial network. In this network, the generator attempts to fool the discriminator into thinking the images it produces are real, and the discriminator tries to overcome the generator's efforts. This struggle causes them both to learn to perform better, eventually improving the generator's efficacy so that its eventual output looks good.

Database and Database CASE tool:

A proprietary dataset of multi-labelled images of buildings (meaning that the images corresponded to multiple categories at once) was necessary to train the neural networks. This was done through a combination of conventional dredging (where appropriate images are extracted from the Internet) and model autophagy (where autonomous tools are used to bulk out a dataset using data generated by an existing model).

We generated our autophagy CASE tool to extract images from Midjourney and created a 2000-image high-resolution dataset to achieve this.

Training function and optimisation CASE tool:

To run, our neural networks needed to be bound to a training function, which propagates 128 image-sized noise matrices through the generator, then separately both the output of the generator and 128 real images from the dataset through the discriminator before backpropagating the discriminator and using the loss of the discriminator to backpropagation through the generator. (then does this again and again until every image in the dataset has passed through the discriminator 100 times)

However, to address the problem of hyperparameter optimisation non-determinism (meaning that the values of certain seemingly arbitrary values within the network have a huge effect on it and no algorithm exists to pick the right ones effectively) we also created an optimisation case tool which uses distributed computing to train the model multiple times simultaneously with different hyperparameter values, then use the results to estimate likely better hyperparameters, before training again, and so on until hyperparameter values converge.

Website:

To make our model more accessible to non-technical users and fulfill the need for our project to have a GUI, we generated a simple website (hosted on the Firebase) using hand-coded HTML and CSS. The website is bound together using XML and can use JavaScript to send user-generated textual prompts to a trained version of the model stored in a separate location.

Extra features:

Originally, we assumed the system's main components would be the neural networks and their corresponding training loop. However, the difficulty in constructing an AI image generator turned out to lie less in the model construction and more in acquiring an appropriate dataset and choosing appropriate hyperparameters.

Therefore, we amended and extended our original specification to include a proprietary multi-label dataset (as no appropriate dataset existed) and a system to optimize the model's hyperparameters autonomously. Both deliverables were completed.

Evaluation:

Overview:

We have been reasonably successful in creating a proof of concept for a larger image generation model. As a team, we have shown great willingness to expand our expertise and work overtime to improve the project.

All of the components of the system work, but unfortunately, we fell down slightly on the integration, as we failed to understand the time it would take both to produce a functional version of the neural network models and to train them once finished and as such were only able to train the optimised architecture for 200/500 epochs before submission. In the future, we want to acquire more processing resources to be able to run training (and therefore hypertuning) faster and commit more time to feasibility research. However, we feel that we have shown an understanding of the neural networking procedure and developed a technically functional model that could be made commercially viable through refinement.

System strengths:

One of the system's core strengths is its ability to process classified distributions. However, I think this is augmented by the fact that it has shown itself to be able to learn to an acceptable standard in less than its complete training time (as we can see from its creation of human-recognisable images only 200 epochs into its 500 epoch training scheme), as well as its ability to function reasonably well when its target distribution is changed, as can be seen from its ability to produce acceptable images in response to the rock-paper-scissors dataset, as well as our dataset.

Another strength of the system is the fact that our dataset is specially crafted for our purposes and conditioned on carefully curated prompts. This is good because, due to the overhead involved in the creation of image datasets, many similar solutions are forced to use datasets which either do not reflect the exact kind of images that developers originally wanted to create or which inadvertently contain artifacts (or other data poisoning) which may affect particular network architectures adversely.

Another strength of the system is that it is modular. So, we were able to develop multiple elements of the system in parallel (saving time) and also offer our database generator and our optimisation framework as distinct case tools (as they are designed to couple loosely with our model and can, therefore, be just as easily applied to other solutions), thus giving us a whole new user type.

Finally, the solution is based on largely experimental technology (the CGAN architecture was only invented in 2017), which gave us an extra challenge to research the underlying mathematics (thus improving our scope and giving us the opportunity to develop research skills and gain experience in an unusual but highly sought-after specialism).

System weaknesses:

One major weakness is that the system does not produce images as quickly as we had hoped. In fact, it takes 60 seconds (instead of 20) to generate a single image and up to 75 seconds via the website.

A second weakness is that the model may only respond to prompts with two overlapping labels. However, this is due to the time and computational power required to generate images corresponding to many well-defined classes necessary for triple overlap.

A third weakness is that the model only accepts integers corresponding to embedded combinations of labels as input (instead of text). As discussed in the future developments section, we would want to address this later using a fuzzy logic analyser.

Finally, the model only operates on a few classes, which we would again fix by expanding the dataset in the future.

Teamwork evaluation:

For the most part, our team has functioned well. We began by agreeing that we wanted to work on an AI project, then did extra research (and often developed new expertise) to make this a reality.

Throughout the process, we have had weekly meetings to update each other on team progress, and everyone has always been aware of their responsibilities each week.

Beyond this, we have extensively used Github’s documentation and version control provisions to assist in debugging, carefully catalogued academic references, and made notes on our research.

Equally, team members have agreed upon weekly assignments in meetings and largely accepted them, listening when team members raised concerns in the group chat (and quickly adjusting to deal with potential errors).

Finally, many team members have been willing to go above and beyond (often fixing problems as they came up or assisting with others' work after completing their own).

However, there were some issues within the team:

A. We left it too late to create standards for terminology use, which caused Matt to try to implement object hitboxing on the database images when, in fact, we wanted images sorted into categories. This was all due to a misunderstanding of the meaning of the term “labelling.”
B. In several instances, Stephen, Wilf, and Emmanuel updated each other’s code to integrate theirs and failed to inform or engage their counterparts, leading to extra debugging of incorrectly updated code.
C. Meshari and Fahad opted to design the website. Still, they did so without checking that they were the most qualified to do so, where in fact, Matt had the most web development experience, and labour could have been saved at the start of the project if they had swapped roles for the first few weeks.

Future Developments:

Immediate fixes:

- Choose a new filter size and stride (for the transpositional convolution layers), which prevents overlap and thus reduces the incidence of checkerboard artefacts. (or if this is not practicable (due to such values damaging learning, replace the transpositional convolution layers with a linear resizing operation coupled with a conventional convolutional layer)
- Acquire a more powerful computer to run the model, thereby reducing the time to generate images.
- Perform additional stress testing on the website.
- Refactor our code to reduce the delay in starting model training.
- Update the optimisation CASE tool to utilise Cuda to run the tool in multiple GPU strands.
- Improve the aesthetic design of the website.
- Add a fuzzy logic circuit to allow prompts to be passed as English language tests and parsed to target classes.
- Add login features to the website, allowing users to save their favourite images.
- Increase the number of supported classes in the dataset by running the autophagy tool.

Long-term fixes:

- Contract additional support to manually dredge the internet for appropriate database images.

- Acquire a Nvidia AI resourcing chip for optimisation.
- Move the website to a commercial hosting location with greater bandwidth.
- Host the model on multiple computers to allow more users to operate it simultaneously.
- Dramatically increase the speed of the autophagy tool to allow for enough supported labels to support higher dimensionality intersectional labels like (english-baroque-abby-granite-1742-foreboding-midday-plan_view)
- Apply feature hitboxing to the dataset so that prompts can include feature requests like (french_spire, or rose_window)
- License a large language model to parse 2 paragraph prompts into highly complex weighted class labels.

BCS compliance:

Public interest:

A. Regarding this point, the images we manually dredged were made readily available on Pinterest and are taken of private property under release (and therefore have no copyright or privacy implications).
B. Regarding this point, our automated images come from an open-source generator, and we are therefore entitled to use them in any context.
C. Discrimination is not a factor in this context since the infamous AI bias problem cannot cause harm here due to the lack of images of humans or decisions impacting humans made by the product.
D. Regarding this point, we have created a user-friendly interface to ensure that non-technical users can access our product.

Professional competence and integrity:

A. Regarding this point, we have completed our specifications competently and, therefore, have undertaken the work we can do.
B. Regarding this point, we have been transparent about our team's failings and admitted that we cannot produce a fully commercially viable product now.
C. Regarding this point, members of this team have done extensive research into (and gained new skills in) neural networking, distributed computing, and data science, as well as carefully complying with industry standards (as failing to do so can cause AI projects to fail catastrophically).
D. Regarding this point, the only relevant legislation is the Data Protection Act, which we have complied with, as can be seen in subsection 1.
E. Regarding this point, team members have continually assessed each other's work, and we have considered feedback from our requirements submission.
F. With respect to this point, our system has very little capacity to promote misinformation or defame anyone; however, our website reminds users that returned images are AI-generated and may contain false or offensive imagery.
G. With respect to this point, we have made no bribery offers and have received none.

Duty to relevant authority:

A. With respect to this point, our submissions have been in the format requested by the university; otherwise, we have only been asked to follow the instructions for the BCS code itself.
B. With respect to this point, when part of our first submission was flagged as AI-generated, we reminded team members not to use generative tools. We began checking our work to ensure it would not appear AI generated due to unusual use of the English language.
C. Regarding this point, we agree to respect the judgment of this university's employees regarding the quality of our work.
D. Regarding this point, all information pertaining to this project has been shared only with the people involved and will only be shared with our assessors in the future.
E. With respect to this point, we have provided evidence of the passage of software tests, and have taken steps to explain the highly technical elements of our project in a way understandable to those not conversant in machine learning.

Duty to the profession:

A. Regarding this point, we have attempted to maintain the appropriate level of reverence in our communications and present our work in a manner that befits the field's professional reputation.
B. Regarding this point, we spent much time designing, testing, and redesigning our communication processes and project management software in the first few weeks.
C. With respect to this point, our project does not pertain to the reputation of BCS in a greatly different manner or extent than it does to the profession's reputation, and those we feel are justified by point A of this subsection.
D. With respect to this point, we have attempted to be respectful and supportive of each other's ideas, feelings, and working habits.
E. Regarding this point, many team members have shared advice about work ethics and practices with each other outside of project communications.

Bibliography:

Dataset Code References:

- Igolaizola, (28/02/23) BulkAI. Available at: https://github.com/igolaizola/bulkai (Accessed: 19 April 2024). - Used for ***208-Project/Dataset/generated_ images/codeGenerate Images.py***
- Singh, A. (05/11/21) Pintirest scraper. Available at: https://github.com/iamatulsingh/pinscrape (Accessed: 19 April 2024). Used for ***208-Project/Dataset/Pinterest Scraper/Pintirest image scraper.py***
- *Pillow Developers, Pillow Documentation. Available at: https://pillow.readthedocs.io/en/stable/ (Accessed: 19 April 2024). Used for -* ***208-Project/Dataset/Pre Processing/code/pre_processing.py***

Website Code References:

- Tech2, (28/05/2023). JavaScript Project | Image Search App With JavaScript API. [Online]. Available at: https://www.youtube.com/watch?v=oaliV2Dp7WQ.(Accessed: 30 March 2024).***208-Project/Website/Public v3/JS/Script.js***

- CodingNepal, (24/08/2019). Animated Login Form using HTML & CSS only | No JavaScript or jQuery. [Online]. Available at:https://www.youtube.com/watch?v=eeHqZeJ9Vqc.(Accessed: 9 May 2024). Used for ***208-Project/Website/Public v3/CSS/loginstyle.css***
- Code Commerce, (3/01/2023). Build A Responsive Sidebar Menu in HTML, CSS, & Javascript - Beginner. [Online]. Available at: https://www.youtube.com/watch?v=uy1tgKOnPB0.Accessed: 9 May 2024. Used for ***208-Project/Website/Public v3/CSS/style.css*** and ***08-Project/Website/Public v3/pages/home.html***
- Singh, A. (05/11/21) Pintirest scraper. Available at: https://github.com/iamatulsingh/pinscrape (Accessed: 19 April 2024). Used for ***208-Project/Dataset/Pinterest Scraper/Pintirest image scraper.py***

Neural network references:

- Sharma,A (12/07/21) CGAN tutorial, available at: Conditional GAN (cGAN) in PyTorch and TensorFlow (learnopencv.com) (used to familiarize myself with keras NN functions)
- Brownlee,B (12/09/19) tips for neural network stabilization, available at Tips for Training Stable Generative Adversarial Networks - MachineLearningMastery.com (used to identify the need for batch normalization and Relu)
- Radford,A|Metz,L|Chintala,S. (2016) Unsupervised learning with CGANS, available at 1511.06434.pdf (arxiv.org) (used to clarify why greater numbers of residual block are important)
- Dosovitskiy, A|Springenberg,J|Tatarchenko,M|Brox,T. (02/08/17) learning to generate chairs with GANS, available at 1411.5928.pdf (arxiv.org) (used to understand the mathematics of convolution, and to justify its use as the core of the models)
- Srivatsavaya,p. (04/10/23) Flatten layer, available at Flatten Layer — Implementation, Advantage and Disadvantages | by Prudhviraju Srivatsavaya | Medium (used to determine the need for post processing in our models)
- Sugawara,Y|Shiota,S|Kiya,H. (19/02/19) checkerboard artifacts, available at Checkerboard artifacts free convolutional neural networks | APSIPA Transactions on Signal and Information Processing | Cambridge Core (used to identify that checkerboard artifacting was occurring in our original CGAN implementation)
- Gulrajani,I|Ahmed,F|Arjovsky,M|Dumoulin,V|Courville,A. (2017) training of Wasserstein GANs, available at Improved Training of Wasserstein GANs | Papers With Code (used to learn the structure of WGANs to support our first hand attempt to produce a WCGAN-GP to address artifacting)