Bespoke data augmentation and network construction enable image classification on small microscopy datasets

Ian Groves[1,2], Jacob Holmshaw[1], David Furley[1,2], Benjamin D. Evans[3], Marysia Placzek[2], Alexander G. Fletcher[1,*]

[1] School of Mathematics and Statistics, University of Sheffield, Hicks Building, Hounsfield Road, Sheffield, S3 7RH, UK

[2] School of Biosciences, University of Sheffield, Firth Court, Western Bank, Sheffield, S10 2TN, UK

[3] Department of Informatics, School of Engineering and Informatics, University of Sussex, Falmer, Brighton, BN1 9RH, UK

[*] E-mail: a.g.fletcher@sheffield.ac.uk

ABSTRACT

Deep learning is a powerful tool for image classification. However, training deep learning models often requires large datasets that are not always available, especially in biological research. Here we leverage a small dataset to train an accurate classifier by employing various data augmentation regimes and Bayesian optimisation. We first generate a fine-grained morphological sub-stage profile of the Hamburger-Hamilton developmental stage 10 chick embryo, finding an asynchrony between somite number and brain morphology. We then leverage our domain knowledge to implement data-driven pre-processing steps, before training a deep convolutional neural network to classify our sub-stages. We find that augmenting our images with a combination of rotation, blur, shear, and cutout transformations is effective in training a high classification accuracy network on a small microscopy dataset. Ensuring reliability of our classifier with saliency analysis, we gain insight into the efficacy of different data augmentation techniques, and identify class-specific features. In summary, we apply domain expertise to design effective data augmentation regimes for training a neural network based image classifier on a small, specialised dataset, classifying the posited developmental sub-stages with up to 90.9% accuracy.

INTRODUCTION

Deep neural networks (DNNs) are used extensively to extract meaningful information from images. Their application includes object localisation, segmentation, and image classification. Deep learning (DL) is increasingly becoming embedded in biomedical research (1), and DNNs have already been used to powerful effect in classifying histological samples from potential cancer patients (2), denoising image data (3), and segmenting objects and regions of interest (4,5). A common maxim for training DNN-based image classifiers is that performance improves with increasing amounts of training data (6,7). However, in some application areas – including many in the biomedical/biological field – it is not always practical to generate large amounts of high-quality training data due to cost, time, or ethical considerations. In such cases, it is important to maximise the utility of limited data when training DNNs, but how best to do this remains an open question. Here we address this in the context of microscopy images used for developmental biology research.

The employment of DNNs in analyses of in vitro fertilisation (8) and cell cycle (9) (reviewed by (10)) first pointed to the power of this approach in the field of developmental biology. In this field a particular challenge is to determine how far an embryo, or embryonic region, has advanced along its developmental trajectory: the embryo's 'developmental stage', and in a recent study (11), DNNs were leveraged to classify explanted zebrafish tailbuds into developmental stages (the zebrafish tailbud is a model for posterior spinal cord growth). In contrast to many developing systems, zebrafish embryos can be obtained in large numbers, and in this study, classifiers were trained on primary images of 2 and 3-D datasets. Bayesian optimization was then used to develop the network architecture, and the success of the algorithm was judged through classification accuracy. While this study provides proof-of-principle for the ability of DNNs to accurately classify development stages, in general, most developmental stage datasets are small and difficult or costly to obtain, when compared to zebrafish, and certainly when compared to the large, readily available, general-purpose datasets such as ImageNet (12). Further, this study did not test the DNN's ability to classify unseen or out-of-distribution data – its capability to generalise (13). This aspect of the use of DNNs in developmental biology is therefore yet to be fully leveraged, and an important goal is to determine the techniques/approaches that best improve generalisation.

When large datasets are not available, data augmentation (through transformations such as rotations) is a particularly useful technique for expanding the training regime, with the intention that a classifier will learn the class-defining features of the image, and will disregard irrelevant features, including image acquisition artefacts (14). Augmentation transformations include those that are 'label-preserving', i.e preserve the features of the image that originally assigned it to a given class. In addition, new data augmentation methods have been developed that only partially preserve labels, for instance, an approach where a 'cutout' is used to mask a random portion of the image (15). This approach reduces the classifier's reliance on those masked parts of the image. As a result, the classifier is encouraged to learn more general features, thus becoming more robust and less sensitive to artefacts in the training dataset.

Another augmentation method that partially preserves labelling are Möbius transformations, which are bijective conformal mappings that preserve angles (16). These transformations may be particularly effective in accounting for user error in microscopy image acquisition. For example, if a user damages the sample during acquisition, Möbius transformations may be able to emulate this. By normalising the user error feature space, the DL algorithm is discouraged from learning these erroneous features.

Besides data augmentation, classifier performance may be improved by optimising parameters of the DNN that are set prior to training. These so-called hyperparameters include the number of computational units (neurons) in the network, and the rate at which these units update – the learning rate. Hyperparameter optimisation is typically carried out via systematic (17) or random (18) testing of values. More recently, Bayesian optimisation techniques have been leveraged, with a surrogate probability model used to inform which values are tested (19). An open question then, when training DNNs on microscopy images, is how to exploit data augmentation and network optimisation techniques to best exploit small datasets?

Finally, saliency maps show which parts of an image the classifier is using to make a classification (20), assessing both the efficacy of data augmentation, and a DNN's capability to generalise. In this way, saliency maps enable insight into the 'black box' approach of using a high accuracy DNN classifier, an important step in increasing uptake of DNN use in developmental biology. Here we assess the impact of data augmentation and network

optimisation on maximising the effective usage of small datasets, and generate saliency maps, with the aim of classifying distinct developmental stages in the chick brain.

Chick embryos have been used to drive our understanding of early brain development (21), but are difficult to obtain in large numbers. In the chick, the rate of development is sensitive to both intrinsic and extrinsic factors (for instance, temperature) so developmental staging is based on morphology, not chronological age. Developmental stages are made with reference to the Hamburger-Hamilton (HH) 1955 staging guide, which assigns embryos to stages between HH1 (newly incubated egg) through 46 (newly hatched chick) (22). This classic staging guide is based on numerous morphological metrics, but for early stages when the brain is forming, relies on the number of somites – segmented pieces of tissue that are laid down at highly regular intervals on either side of the spinal cord (23). However, careful observational studies show that the rate/timing of development of different body structures, including the brain – is subtly uncoupled from somite number, and increasingly, researchers are attempting to develop staging classifiers for distinct embryonic regions (23–27).

One part of the brain that is highly conserved between different vertebrates is the hypothalamus. This region of the brain governs homeostasis, including energy balance and stress regulation. Importantly, studies show that disordered hypothalamus development promotes adult diseases, including obesity and stress related disorders (28–31), and increasing numbers of studies are therefore beginning to focus on hypothalamic development. Recent studies in the chick show that the developing hypothalamus undergoes a significant expansion over HH10 (a period that encompasses ~5 hours), accompanied by subtle changes in its morphology (32,33). Molecular studies show that expansion of the early hypothalamus is accompanied by the first signs of its regionalisation into distinct functional domains (33). This means that, for the accuracy of future experiments, it is important to develop a finer-grained classification of the HH10 hypothalamus. However, while highly-trained developmental neurobiologists are able to sub-classify the HH10 hypothalamus based on the broader morphology of the brain, those with less experience are unable to do so with confidence.

Here, we leveraged domain expertise to develop a reproducible, rapid, and automatable classifier that subdivides developmental stages of the HH10 brain, including the

hypothalamus. To do so we developed a deep convolutional neural network (DCNN) based image classifier. We tested both the efficacy of widely used and bespoke network architectures, including a large variety of data augmentations to mitigating the effects of small, noisy microscopy datasets. From training a classifier to extract identifying features of microscopy images, we develop an important tool for conducting experiments on chick embryos by accurately classifying the developing hypothalamus, at HH10, into three sub-stages. We then investigate the classification process using saliency analysis, finding novel sub-stage specific image features.

RESULTS

Somite number does not explain variation in HH10 brain morphology

Our visual inspection of images suggested that the chick HH10 brain could be divided into three sub-stages, according to criteria based on the morphology of the developing prosencephalon (the forebrain), which we termed 10.1, 10.2 and 10.3. Three reference images illustrating these sub-stages are shown in **Fig 1A.** At 10.1, the developing brain is kite-shaped, the optic vesicles (developing eyes) are shallow, and the angle of the prosencephalic neck, the anatomical landmark where the forebrain meets the posterior brain, is obtuse (**Fig 1A, Fig 1B iii**, left columns). At 10.2, the developing brain is hammer-shaped, the optic vesicles are more pronounced, and the posterior prosencephalon-prosencephalic neck are aligned (**Fig 1A**, **Fig 1B iii**, middle columns). At 10.3, the angle of the posterior prosencephalon-prosencephalic neck is acute, and there is a significant amount of curvature in the optic vesicles (**Fig 1A**, **Fig 1B iii** right columns). The overall shape profiles that we picked out for our sub-staging, including the prosencephalic neck, are schematised in **Fig 1B.**

We next aimed to understand the correlation between our sub-stages and the classic Hamburger-Hamilton staging chart (22), where developmental stage is judged on the basis of somite number: HH10 embryos have 9, 10 or 11 somites, and are traditionally termed HH10- (9 somites), HH10 (10 somites), or HH10+ (11 somites) (22). First, we quantified the number of somites at each sub-stage (**Fig 2A**). We observed that embryos at each sub-stage (10.1, 10.2, 10.3) can have a wide range of somites (8-12 somites) (**Fig 2B**). We

also found significant differences in somite number between 10.1 and 10.3 (*** p=0.0002), and between 10.2 and 10.3 (* p=0.015), but found no significant difference in somite number between 10.1 and 10.2 (p=0.079) **(Fig 2B)**. In conclusion, somite number is a poor correlate of chick HH10 brain development, underscoring the need for an alternate fine-grained classification approach at this developmental stage.

Unsupervised learning via *k*-means clustering supports three sub-stages, but performs poorly as a classifier

We next sought to determine whether the choice of three sub-stages, motivated by our expert visual inspection, could be supported through clustering approaches. Using a similar approach that has successfully been applied to cell phenotype classification in histology images (34), we used unsupervised learning via a combination of principal component analysis (PCA) and *k*-means clustering. Thus the intent here was to: (i) determine an appropriate number of classes for a further neural network based classifier; and (ii) test the efficacy of alternate methods of classification.

A training dataset of 151 images was labelled according to the reference images shown in **Fig 1A** (54 images at 10.1, 55 at 10.2, and 42 at 10.3). We then used PCA before *k*-means clustering for dimensionality reduction, which is usual practice to ensure comprehensible and robust clustering (35). To empirically determine an appropriate number of principal components, we generated a scree plot (**Fig 3A**). Here, we calculated the total variance explained by reducing our chick embryo sub-stage dataset into increasing numbers of principal components. We found that two principal components explained 81% of the variance in the dataset, and increasing the number of principal components further led to substantially diminishing returns.

Following this, we performed an unsupervised machine learning analysis via *k*-means clustering. Clustering through *k*-means requires the number of clusters around which the data will be grouped to be determined *a priori*. To determine an optimal number of clusters, we used the elbow method (**Fig 3B**): we applied *k*-means clustering to the dataset, with between 1 to 10 clusters, and selected the ideal number of clusters as the inflection or 'elbow' point in the within-cluster sum of squares (WCSS), where increasing *k* begins to show diminishing returns in WCSS. The elbow point was reached at three clusters.

Following this, we visualised the *k*-means clustering with *k* = 3 (**Fig 3C**). From this, we found substantial differences in the number of images within each cluster. Between the three clusters, cluster 1 contained 84 images, and clusters 2 and 3 contained 18 and 49 images respectively. (**Fig 3D**). We assumed that the sub-stage that was most represented in each cluster is the true classification for that cluster. Thus, sub-stages 10.1, 10.2 and 10.3 corresponded to clusters 2, 1 and 3, respectively (**Fig 3D**). Therefore, the classification accuracies achieved by PCA followed by *k*-means clustering were: 10.1: 38.9%; 10.2: 41.7%; 10.3: 44.9%.

As *k*-means based classification is unsupervised, it is unlikely to group data in the same way as a human observer. Looking beyond the classification accuracies, the distribution of images between clusters is skewed. Our '10.3' cluster, cluster 3 performed best, but the accuracy achieved and the distribution of images in the clusters suggest that this clustering is not effective for determining sub-stage (**Fig 3D**). These low accuracies and unequal distributions suggest that at least one feature is being used for clustering which is irrelevant to the true class. Whilst we confirmed that three sub-stages is an appropriate number of sub-stages to use, we next aimed to substantially improve the classification accuracy achieved by using DCNNs.

A DCNN trained using the ResNet50 architecture achieves a classification accuracy of up to 75% using a combined data augmentation regime of rotation & Gaussian blur

As our dataset is relatively small (151 images), it is important to fully utilise every image. There are two useful strategies when dealing with small datasets. First, the dataset can be augmented with various transformations, which can expose the network to features that are unimportant for classification, but may have a skewed distribution throughout the dataset, e.g. subject orientation. However, the various benefits and effects of applying different augmentations to microscopy data are relatively unexplored. As such, we sought to understand which augmentations would perform best on our dataset.

Another useful strategy is to employ an exhaustive type of validation, *k*-fold cross validation. This contrasts with the commonly used hold-out validation, where 20% of the dataset is unavailable for training. We therefore used *k*-fold cross validation, with *k* = 10, resulting in 15-16 images per fold, which uses the whole dataset for training. Here, ten (equal to *k*)

DCNNs were trained, where nine partitions (folds) of the dataset were used as training data, and the remaining fold was used for validation. Each network was trained whilst varying the validation fold and training data accordingly.

First, we examined the potential for training our classifier using the widely used image classification network architectures from InceptionV3 (36) and ResNet50 (37). These two neural networks have achieved high classification accuracies on ImageNet, a large database of over 14M images, typically used for comparing neural network image classifiers. As ResNet50 and InceptionV3 are well established in image classification tasks, we sought to understand to what extent these architectures could be leveraged for small datasets of microscopy images.

Designing an effective data processing and augmentation approach requires careful consideration of the data. Within the dataset, any obviously skewed distribution of features is a candidate for an augmentation regime to normalise that distribution. Of the various data augmentations that are easily implemented, we reasoned that augmenting with rotations, and normalising the histograms of the images as a pre-processing step would be a useful baseline. This is intuitive, as microscopy images of developing embryos are often oriented in various ways. Histogram normalisation was also a natural choice for a baseline processing step, as microscopy images can greatly differ in brightness.

Following this, we modified our original dataset (**S1 Fig A**) with a variety of augmentations. (**S1 Fig B**). We then trained DCNNs on these different data augmentation regimes, using the InceptionV3 and ResNet50 architectures, and measured the classification accuracy. The results of training these models can be seen in **Tables 1 & 2**. In general, InceptionV3 did not perform well, with similar averages across various data augmentations, and a maximum validation accuracy of 60.5%. This is marginally better than the results seen through *k*-means classification, where the highest accuracy was 44.9% (**Fig 3D**), however we note the average validation accuracies achieved by InceptionV3 are in the range of 44-46%. With ResNet50 training, we saw a marked improvement in validation accuracies over InceptionV3 and the *k*-means clustering accuracy across all augmentations except for crop, which only improved by an average of 3%. Our baseline & Gaussian blur augmentation regime had the highest average validation accuracy, with 69.2%, and highest fold accuracy of 75.9%. Additionally, baseline & Gaussian blur achieved the second lowest standard deviation, an

important metric to consider in light of a limited dataset (**Table 2**). Taken together, these results suggest that transfer learning is not always suitable for image classification of small microscopy datasets. However this poor performance can be substantially ameliorated by careful data pre-processing, with rotation & blurring augmentations.

A bespoke neural network classifies an augmented chick embryo dataset with up to 90.9% accuracy

ResNet50 has been used successfully for image classification on datasets other than ImageNet, for example on ChestX-ray14, a dataset of medical X-ray images (38). ChestX-ray14 is substantially larger than our dataset, comprising 112,120 images. Therefore, considering that we did very little fine-tuning of the hyperparameters of ResNet50, and that ResNet50 was constructed for large datasets, ResNet50 performed surprisingly well (up to 75.9%) when trained on our data. However, the validation accuracy leaves room for improvement so we set out to understand whether a bespoke network would achieve higher classification accuracies than ResNet50. This has the benefit of incorporating domain expertise by being specifically designed for our problem.

To this end, we constructed a model with a wide, block style architecture (**S2 Fig**) following similar construction to VGG-16, a model architecture which has been successful in image classification (39). We used a mixture of Bayesian optimisation (**S3 Table**) and examples from the literature to determine the hyperparameters (see materials and methods for details). We then used this architecture to test which are the most useful and robust augmentation regimes, which are also resilient to small datasets. The results for our model are shown in **Table 3**. Our baseline usually performed well, surpassing our best ResNet50 results with an average validation accuracy of 76.2%. However, for each fold, one of the various augmentations always had higher validation accuracy than the baseline. The poorest performing augmentation was baseline & crop, with an average validation accuracy of 47.1%. The best performing individual augmentation was baseline & Gaussian blur, with the highest average validation accuracy of 80.5% and the lowest standard deviation (excluding crop, which performed substantially below baseline). Baseline & cutout provided the best individual fold validation accuracy, of 91.7% at fold 9 (vs. Gaussian blur 88.9%). Despite baseline & cutout achieving the highest individual fold validation accuracy, we conclude that

baseline & Gaussian blur overall performed best, as it achieved the highest validation accuracy on folds 4-7.

We also tested the efficacy of Möbius transformations, which have been previously found to be successful as an augmentation in general image classification (16). Yet, to our knowledge, Möbius transformations are untested for microscopy image classification. We reasoned that Möbius transformations could be a sensible approach, as they would introduce the DCNN to artefacts common in microscopy imaging, e.g. tissue bending during sample preparation. However, we found that our baseline and Möbius transforms performed more poorly than the baseline alone, and also that sparse inclusion of Möbius transformations alongside Gaussian blur decreased validation accuracy (**S1 Table**).

The *k*-fold cross validation results (**Table 3**) show that there are parts of our dataset that lead to low accuracies. This is likely due to artefacts introduced in chick dissection or preparation for microscopy imaging, e.g. due to deformation/tearing of the fragile brain tissue. As such, we sought ways of mitigating this invariance. Whilst the good performance of our baseline is intuitive, both the shear and Gaussian blur regimes also achieved high validation accuracies. In the case of shear, we reasoned that this could reflect increasing the robustness of the network by introducing it to distortions similar to tissue deformation in sample preparation. Similarly, Gaussian blur could increase robustness by simply introducing the DCNN to data that resembles out-of-focus images, which are a common artefact in microscopy.

To test whether Gaussian blur is acting in this way, we applied the inverse operation by using a sharpening filter on our images as a pre-processing step. We then augmented our data with the baseline augmentation of rotation (**Table 3**, 1 Sharpened). We found that a simple sharpen filter increased average validation accuracy from 76.2% (baseline) to 78.2% (baseline sharpened). Further, we saw that poor performing baseline folds e.g. 2 and 8 had substantially higher validation accuracies than the baseline alone. However, the results from applying a sharpen filter as a pre-processing step do not explain the entire discrepancy in validation accuracies between our baseline and Gaussian blur. This suggests that Gaussian blur is not only introducing the DCNN to out-of-focus images, but could be masking tissue tearing artefacts through smoothing of geometric lines or curves in the tissue structure. Moreover, these augmentations may be assisting the network to focus on a true sub-stage characteristic, rather than any features introduced by biological inter-sample variation. This

suggests that these poorly scoring dataset partitions contain images with features that lead to an inaccurate classification, but this can be mitigated with reasoned pre-processing and augmentation.

Since different augmentations had scored highest on different folds, we next tested different mixes of these augmentations as an approach to reduce this variability (**Table 3**, 1 + 2,4,5 RC). We identified an informed combined regime as a useful approach when dealing with small datasets (where variability is likely to be high). For example, folds 2 and 8 were the two partitions of the data with the lowest average validation accuracy across all augmentations. In both cases, mixing the baseline with shear, Gaussian blur, and cutout resulted in substantially higher validation accuracies. This suggests that our combined set of augmentations allowed the network to learn features of images which are 'difficult' to learn.

In summary, we find that training a bespoke neural network suits our problem well, and that intuition-motivated, reasoned augmentations enhance the classification accuracy of our network, achieving >90% validation accuracies, despite a small dataset.

Saliency maps identify differences between classifiers when trained on differently augmented data

We next performed saliency analysis (20) to determine to which region(s) of our images our classifiers were most sensitive. We selected the baseline model, and a range of models trained on different augmentation regimes (**Table 3**), and visualised their saliency maps for test images of each sub-stage (**Fig 4Ai-vi**, **S3 Fig Ai-vi**, **S3 Fig Bi-vi**). The 10.2 saliency maps indicated that our baseline model focused a large amount of attention on the prosencephalon, but also some scattered attention in more peripheral parts of the image. (**Fig 4Ai**). This was also reflected in the 10.1 maps (**S3 Fig Ai**), but less so for the 10.3 maps (**S3 Fig Bi**). Models trained on baseline plus Gaussian blur (**Fig 4Aii**), cutout (**Fig 4Aiii**), and shear (**Fig 4iv**) reduced this scattered attention substantially, causing the classifier to hone in on several focal points (compare **Fig 4Ai with Fig 4Aii-iv**). Similar effects of these augmentations were seen in the 10.1 and 10.3 maps (**S3 Fig Aii-iv, S3 Fig Bii-iv**). Gaussian blur also introduced a focus on the 10.2 prosencephalic neck (**Fig 4Aii**). We included crop as a negative control, as it performed substantially worse than all other augmentations (**Table 3**). Here, crop either produced very little focus (**Fig 4Avi**) or scattered attention maps

throughout the entire image (**S3 Fig Avi**, **S3 Fig Bvi**). These results confirm that the choice of augmentation results in different image features being picked out by a classifier, but also that biologically relevant features e.g. the prosencephalic neck can be focused on as a result of Gaussian blur.

We next tested the DCNN trained on our combined regime, applying Gaussian blur, cutout, and shear over the baseline at random for each training image. This augmentation regime produced a very similar average validation accuracy and standard deviation score to baseline with Gaussian blur (**Table 3**, the highest scoring individual augmentation). The saliency maps generated by this model have large focal points on the centre of the prosencephalic neck (**Fig 4Bi**, magenta arrow) and the junction of the prosencephalon-prosencephalic neck (**Fig 4Bi**, white arrow). We also saw a 'combined map' effect, where the baseline with random augmentation regime produced focal areas which resemble larger versions of focal points from the other augmentation regimes alone (**Fig 4A**, **S3 Fig Av, Bv**). In particular we found clear differences in the focal points paid to each class. For 10.1, the combined augmentation regime-trained DCNN focused on the prosencephalic neck and the shape of the prosencephalon (**S3 Fig Ci**, magenta and white arrow respectively), while for 10.3, focal points were more posterior and confined to the neuromeres (**S3 Fig Cii**, white arrows). The 10.2 sub-stage shown in **Fig 4B** contrasted with these, showing more focal points spread between the prosencephalon and more peripheral regions, possibly reflecting that this sub-stage shares similarities with both 10.1 and 10.3, as it is situated between them in developmental time.

From examining these saliency analyses, we can unpick the decision making of models trained on different data augmentations. For example, the data were labelled according to prosencephalic morphology (**Fig 1**). However, our training dataset also included more posterior regions of the embryo, e.g. the neuromeres, which were not explicitly used to label the dataset. Exemplifying the utility of our approach, we found that the sub-stage specific features of 10.3 focused on these more posterior regions (compare **S3 Fig Cii** and **Fig 4B, S3 Fig Ci**), which was surprising, as these areas were not considered in labelling. Thus, training a DCNN to classify these sub-stages provides a deeper insight, as the DCNN is free from human preconceptions about important features.

As described prior, the prosencephalon in these chick embryo samples is a fragile structure, and is prone to deformation and damage, and Shear and Gaussian blur augmentations are likely to ameliorate invariance caused by these issues. However, by expanding a dataset with combinations of reasoned, biologically motivated augmentations we can consolidate the benefits of different augmentations. By doing so, we detract from inter-subject variability, and increase focus on the true sub-stage structure. As a result of this increased focus, we found that our labelling intuitions/criteria were correct about 10.1 and 10.2, but that the most predictive feature of 10.3 embryos was the shape of the more posterior neuromeres. These results suggest that rigorous examination of classifier attention gives insight not only into the effects of different processing/augmentation regimes, but also into the anatomical features of the embryo which most define the sub-stage classification.

DISCUSSION

In this work, our approach was to combine the objective, systematic nature of machine learning approaches with domain expertise. This helped us to design bespoke data processing and augmentation regimes, which in turn allowed us to train an accurate and reliable classifier on small microscopy datasets.

In the current study, clustering confirmed the suitability of three sub-stages. In other developmental contexts more sub-stages may be appropriate, and our approach naturally extends to that. From an experimental perspective, the number of sub-stages is a balance between precision and ease of analysis. The utility of our method is that the number of sub-stages emerge in an objective and systematic way.

Further work may seek to add more classes and thus a more fine-grained binning, however this would require a lot of high resolution image data, which is both experimentally arduous, and increases computational load when training the classifier. Further work could instead look to expand the dimensions of the images used. We used 2-D morphological profiles to train our classifier. Diversifying the dataset with 3-D fluorescent images, which are increasingly used in developmental biology, could provide a more robust, accurate classifier. Additionally, by including data on biological events, e.g. gene expression data, we could couple our sub-stages to a biological mechanism(s). Finally, the classifier could be further

expanded to other developing organs within the chick embryo, where there exists uncoupling of somites from morphology, for example in the developing limb (25).

Previous efforts to classify microscopy images in developmental biology have focused on hyperparameter optimisation (11). In contrast, we have instead focused on exploring different data augmentations, as it is more tractable to fully exploit domain expertise when augmenting data, as opposed to hyperparameter optimisation. Recent theoretical work supports our approach, highlighting the superiority of data augmentation at optimising the learning of DCNNs (40). We saw substantial differences in model performance with different data augmentations, indicating the importance of careful augmentation regime design. We also found that combining individually successful augmentations was an effective method to consolidate the benefits of these augmentations when training our classifier. We therefore conclude that careful analysis of the data and an appropriate augmentation regime should be designed for specific datasets and problems, such as those in developmental biology.

We found that training using InceptionV3 and ResNet50 architectures is not always suitable for all classification problems, specifically on small datasets. However, this is not surprising, as these models were originally built for and trained on ImageNet, which is more general and larger than our dataset. Indeed, other work on biological image classification using small (albeit larger than ours) datasets has encountered issues using a ResNet50 based architecture. Margapuri et al. tested various transfer learning models, including ResNet50 at classifying different bee species (41). They found that ResNet50 overfits substantially on their data, achieving at best a 7.4% accuracy. They also tested InceptionV3 and found similar results to ours, with a maximum accuracy of 44.5%. Whilst these authors did test augmentations including rotation and cutout, they found that the un-augmented data achieved higher accuracies. Considering these and our findings, it is clear that applying widely used image classification architectures alone is not a straightforward solution when dealing with small datasets.

In this work, we have characterised a new staging system for the chick embryo at HH10. We identified three distinct sub-stages of HH10, based on morphology of the developing prosencephalon. We showed that these sub-stages are not explicable by somite number, and therefore somite number is uncoupled from the developing brain morphology (at least at HH10). This result aligns with other studies on different developing tissues. For example,

Musy et al. devised a morphometric staging system of the developing mouse limb, motivated by an asynchrony in somite number and limb bud morphology (25). Additionally, Sáenz-Ponce et al. found an uncoupling between neural development and somitogenesis in various frog species (26). Taken together, our work and these studies find that the number of somites is, at least at certain stages, uncoupled from wider organ development.

Advances in higher-resolution data collection methods (42), including detailed timecourse tracking of embryonic development by light microscopy, necessitate careful consideration of traditional staging guides. Our studies underscore the importance of finding new ways to finely gauge developmental stages in early brain development: we find that the number of somites is out of sync with prosencephalic development. This has consequences for experiments on the development of early brain regions, including the hypothalamus, whose accuracy may be key in developing treatments for various neuroendocrine disorders, such as obesity and chronic stress (28–31)

In this work, we used saliency maps to gain insight into the regions of the images to which our classifier was most sensitive. This work illustrates the potential utility of saliency analysis in helping to interpret classifiers in a way that is understandable to experimental biologists. This approach has been used successfully in other areas of biomedical research, e.g. X-ray images of lung cancer based on labelling provided by radiologists (38,43). Here, the authors leveraged saliency methods to understand the decision making of their classifiers, an important step if DNN based classifiers are to have a role in medical decision making. The use of such methods has the potential to encourage confidence in non-specialists about DCNN-based classifiers. There is also, as we show here, potential for insight into the most predictive features of labelled classes, free from human preconceptions. Therefore, these analyses encourage confidence in the classifier, which will increase uptake by the wider developmental biology field.

MATERIALS AND METHODS

Data acquisition

Ground-truth data used for training and validation of the image classifier comprised bright-field and phase contrast microscopy images of HH10 (9-12 somite) chick embryos, and included both published and unpublished data. Images were acquired using the following microscopes: Olympus BX60, Zeiss AxioImager.Z1 (Apotome.2), and a Leica dissecting microscope at 4x or 10x magnification. Data for training were labelled into three sub-stages, termed 10.1, 10.2 and 10.3, based on our domain expertise, according to the gross morphology of the HH10 brain. Briefly, stages were assigned according to the overall shape of the prosencephalon, the angle of the posterior prosencephalon relative to the prosencephalic neck, and the shape of the optic vesicle. There are 151 images in the dataset, with 54 sub-stage 10.1 embryos, 55 sub-stage 10.2 embryos, and 42 sub-stage 10.3 embryos. The images were JPEG format, varying in resolution, from 270 x 244 to 1000 x 1000 pixels.

Statistical analysis of somite numbers

We did not assume that the variance of somite number with sub-stages is evenly distributed across the sub-stages. Therefore, this statistical relationship was analysed by unpaired *t*-test with Welch's correction.

Clustering analysis

For the clustering analysis, we first reduced the dimensionality of the raw images via principal component analysis (PCA) (44). We determined the appropriate number of principal components to retain by iteratively increasing the number of principal components in the analysis. We then set the number of principal components to 2, the number at which we saw diminishing returns in the amount of variance accounted for by using more principal components (PC > 2). We then performed *k*-means clustering (45), using a similar approach to determine the correct value for *k*. Here we clustered the PCA reduced dataset using increasing numbers of clusters. As the number of clusters increases, the within-cluster sum of squares (WCSS) reduces. We set the value of *k* to 3, the elbow point (46). This is the point above which we see diminishing returns in WCSS reduction by adding another cluster. Both the PCA and *k*-means clustering were implemented in Python 3.7.12, with the library scikit-learn 0.24.1 (47). The plots in Fig 2, and Fig 3 were generated using the plotting libraries seaborn 0.11.2 (48) and matplotlib 3.2.2 (49).

Data pre-processing

The various data augmentation regimes we employed are listed in **Table 4** and visualised in **S1 Fig**. To make the trained model invariant to parameters in the image which were not classifying features (e.g. scale, colour), various pre-processing methods were applied. Images were handled and converted to 200 x 200 pixels using the libraries openCV (3.4.2.17) and pillow (8.3.1) (50,51). 200 x 200 is a sufficiently small size to be easily processed, whilst retaining sufficient spatial resolution to distinguish morphology. Following this, the images were converted to grayscale. In this way, any benefit that additional data augmentations provided could be assessed easily. We normalised the histograms of each image, using the pillow function ImageOps.equalize (51). Equalising the image histograms served to brighten images that were too dark and darken images that were too bright. For the sharpen filter (**Table 3**) we used the pillow function ImageFilter.SHARPEN (51).

Data augmentation

Following pre-processing, we constructed a baseline augmented dataset. For microscopy images, a common user-generated feature during data collection is the orientation of the sample/subject. Thus we included rotation as a baseline augmentation of our dataset, choosing to rotate each image by 36 multiples of 10°. Our rotation method enlarged the images on rotation so as to leave no part of the image cut off. This meant that in addition to rotational and colour invariance, there would also be incorporation of scale invariance into the baseline dataset. All augmentations were implemented with the package imgaug 4.0 (52), with the following methods: rotation: imutils.rotate_bound, shear: imgaug.ShearX, crop: imgaug.Crop, Gaussian blur: imgaug.Gaussianblur, cutout: imgaug.Cutout.

Cross-validation

Initially, we used hold out cross-validation, where a portion of the dataset (in our case, 20%) is held back to be used as a validation set. Here, we trained 10 models for each augmentation regime, reshuffling the training/validation data each time. Our results were substantially variable, e.g. for models trained on our Baseline dataset, the difference between the minimum and maximum validation accuracy was 26.9%. We supposed that the

variability seen during the hold out cross-validation was a product of our small dataset. As 20% of the dataset was being held back for validation, even just a few skewed features present in the training or validation set would affect results. Therefore, we employed *k*-fold cross validation. This ensured we made use of the entire dataset, as well as isolating the portions of the data which were contributing to variable results (53). Briefly, we partitioned the dataset into 10 non-overlapping folds of 15 images per fold. We then trained the network on folds 2-10 and validated with the first fold. Following this, the network was then trained with folds 1 and 3-10, with the second subset used to validate. This training proceeded until all folds had been used. In this way, we averaged performance of our neural network across the entire dataset. The advantage here is that the networks are trained on the entire dataset, as a changing partition of the dataset is held back for testing. We therefore implemented our *k*-fold cross validation with *k*=10 (**Tables 1-3**).

Neural network architecture

We based our bespoke DNN on the Visual Geometry Group (VGG-16) model architecture, which has been highly successful as an image classifier (39) (**S2 Fig**). This involves repeated functional units of the network, often referred to as VGG 'blocks'. These blocks consist of a convolution layer with resolution preservation, followed by a max pooling layer which performs spatial down-sampling by a factor of 2 each time. Our architecture departs here from VGG-16 as we only include one convolutional layer between each max pooling layer. Between the convolutional layers and the max-pooling layers there is an activation function. For this activation function, we used the Rectified Linear Unit (ReLU) which is widely used and is considered best practice (54). As the actual spatial resolution of the data decreases, the number of filters used doubles. This means the first layer that receives the 200 x 200 image input has 16 functional units, which is repeated 6 more times resulting in a final convolutional layer with dimensions 4 x 4, with 1024 functional units. This follows a similar pattern to VGG-16, however the largest convolutional layer in VGG-16 is 512 wide, whereas we extend to a layer of size 1024. Following these blocks, we include a fully connected layer of 1024 units which receives input from each neuron in the last max-pooling layer, followed by a softmax classification layer. Here, our model differs from VGG-16, which uses three wider fully connected layers. All neural networks were built and trained using TensorFlow 2.2.0 (55).

InceptionV3 and ResNet50

We first tested two common neural network architectures, InceptionV3 (36) and ResNet50 (37). First, we randomised the weights, before training for no more than 50 epochs (full training cycles), the exact number varied as we implemented early stopping. Here, we halted training after 10 epochs had passed without a change in validation accuracy over a 0.01% threshold. The number of epochs to pass, and the early stopping threshold, were selected empirically based on the speed at which models that were allowed to run to 50 epochs converged. When this was triggered, we restored the highest scoring weights in training before saving the model. We randomised the neuron weights for both ResNet50 and InceptionV3. Following the typical approach for multi-class classification problems (56), we inserted a classification layer of three nodes, with a softmax activation function as the last layer in the model. Here, the softmax activation will perform the actual classification by transforming the input between 0 and 1. Thus, the outputs of the model are three values which sum to 1, which we interpret as an effective probability of the input belonging to a given class. This was necessary as both InceptionV3 and ResNet50 were initially designed around the ImageNet dataset (1000 classes). We used the optimiser Adam with an initial learning rate of 10^{-4} (57), but decreased this to 10^{-5} after we observed overfitting. However, the value of 10^{-5} is also commonly used for image classification (41,57). The exact number of epochs varied between models, as we implemented early stopping, Hyperparameter fine-tuning was implemented using TensorFlow 2.2.0 (55) and Keras 2.3.0 (58).

Training regime

As above, we trained for no more than 50 epochs, and implemented early stopping. We regularised our network with L_2 regularisation. Also known as weight decay, this technique penalises large weights in the neural network by adding a penalty to the loss function of the neural network (56). The key parameter, λ, is a fraction of the sum of the squared weights of the network. As λ increases, the loss function value increases. Because a neural network is optimised by minimising the loss function, L_2 regularisation encourages smaller weights and thus less complex models. We chose the numerical value of λ that we used in L_2 regularisation through Bayesian optimisation (**S3 Table**) as $\lambda = 10^{-4}$. We also find that this value has been found to be effective in training image classifiers (59). We also utilised dropout, which randomly turns off neurons in a layer at a given rate (60). This serves to

discourage individual neurons from becoming dominant, which in turn encourages a classifier with better generalisability. We added a 20% dropout layer between each convolutional and max pooling layer, and a 50% dropout layer before the final classification; these percentages were also determined through Bayesian optimization (**S3 Table**). We used the optimiser Adam, with a learning rate of 10^{-4}, determined through Bayesian optimisation (**S3 Table**). Here, we set our range of trialled learning rates to test during optimisation (10^{-1}–10^{-6}) according to our previous transfer learning rate of 10^{-5}. Our model has fewer trainable parameters than both InceptionV3 and ResNet50. Due to this, our training accuracy converges slower, and therefore it is unsurprising that our model performs optimally with a faster learning rate than ResNet50 or InceptionV3.

Saliency analysis

We generated saliency maps using the Python package tf-keras-vis 0.8.0 (61). In these maps, image pixels are assigned a colour based on whether they contribute positively (hot colours) or negatively (cold colours) towards the output prediction. In practice, this produces a map of the features of an input the network deems most and least important towards making a prediction. We used the SmoothGrad method, which produces clean saliency maps by adding noise to the input image (62). To understand the decision making of the classifier on real world data, we generated the saliency maps using test images, which were not involved in the training/validation pipeline of any of the models.

Software

The neural networks were built and trained using Python 3.6. The models were trained on a NVIDIA Tesla V100 GPU using the HPC system provided by the Joint Academic Data Science Endeavour (JADE) II. The code and exhaustive list of software versions used in this project is publicly available at https://github.com/ianbgroves/chick_embryo_DCNN_classifier.

AUTHORS' CONTRIBUTIONS

I.G. Project conception, data analysis, programming, manuscript writing, manuscript review. J.H. Data analysis, programming. D.F. Data analysis, programming. B.E. Project conception,

manuscript review. M.P. Project conception, data labelling, manuscript review. A.G.F. Project conception, manuscript writing, manuscript review.

ACKNOWLEDGMENTS

We thank K. Chinnaiya and S. Burbridge for providing unpublished images of chick embryos and for comments on the manuscript.

COMPETING INTERESTS

We have no competing interests.

FUNDING

This work was supported by the Wellcome Trust (grant number 212247/Z/18/Z to M.P.); and the Engineering and Physical Sciences Research Council (PhD studentship to I.G.).

REFERENCES

1. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–44.
2. Iizuka O, Kanavati F, Kato K, Rambeau M, Arihiro K, Tsuneki M. Deep learning models for histopathological classification of gastric and colonic epithelial tumours. Sci Rep. 2020;10:1–11.
3. Krull A, Vičar T, Prakash M, Lalit M, Jug F. Probabilistic Noise2Void: Unsupervised Content-Aware Denoising. Front Comput Sci [Internet]. 2020 [cited 2022 Feb 14];2. Available from: https://www.frontiersin.org/article/10.3389/fcomp.2020.00005
4. Schmidt U, Weigert M, Broaddus C, Myers G. Cell detection with star-convex polygons. In: Medical Image Computing and Computer Assisted Intervention. 2018. p. 265–73.
5. Weigert M, Schmidt U, Haase R, Sugawara K, Myers G. Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy. In: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV) [Internet]. Snowmass Village, CO, USA: IEEE; 2020 [cited 2022 Feb 14]. p. 3655–62. Available from: https://ieeexplore.ieee.org/document/9093435/
6. Thompson NC, Greenewald K, Lee K, Manso GF. The computational limits of deep learning. ArXiv Prepr ArXiv200705558. 2020;
7. Jacquemet G. Deep learning to analyse microscopy images. Biochem. 2021;43:60–4.
8. Khosravi P, Kazemi E, Zhan Q, Malmsten JE, Toschi M, Zisimopoulos P, et al. Deep learning enables robust assessment and selection of human blastocysts after in vitro fertilization. NPJ Digit Med. 2019;2(1):1–9.
9. Eulenberg P, Köhler N, Blasi T, Filby A, Carpenter AE, Rees P, et al. Reconstructing cell cycle and disease progression using deep learning. Nat Commun. 2017;8(1):1–6.
10. Hallou A, Yevick HG, Dumitrascu B, Uhlmann V. Deep learning for bioimage analysis in developmental biology. Development. 2021;148(18):dev199616.
11. Pond AJR, Hwang S, Verd B, Steventon B. A deep learning approach for staging embryonic tissue isolates with small data. PLoS One. 2021;16:e0244151.
12. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L. ImageNet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE; 2009. p. 248–55.
13. Rosin PL, Fierens F. Improving neural network generalisation. In: International Geoscience and Remote Sensing Symposium. IEEE; 1995. p. 1255–7.
14. Simard PY, Steinkraus D, Platt JC. Best practices for convolutional neural networks

applied to visual document analysis. In: Icdar. 2003.
15. DeVries T, Taylor GW. Improved regularization of convolutional neural networks with cutout. ArXiv Prepr ArXiv170804552. 2017;1–8.
16. Zhou S, Zhang J, Jiang H, Lundh T, Ng AY. Data augmentation with Mobius transformations. Mach Learn Sci Technol. 2021;2:025016.
17. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998;86(11):2278–324.
18. Bergstra J, Bengio Y. Random search for hyper-parameter optimization. J Mach Learn Res. 2012;13(2).
19. Shin S, Lee Y, Kim M, Park J, Lee S, Min K. Deep neural network model with Bayesian hyperparameter optimization for prediction of NOx at transient conditions in a diesel engine. Eng Appl Artif Intell. 2020;94:103761.
20. Simonyan K, Vedaldi A, Zisserman A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. ArXiv13126034 Cs [Internet]. 2014 Apr 19 [cited 2022 Feb 17]; Available from: http://arxiv.org/abs/1312.6034
21. Placzek M, Briscoe J. Sonic hedgehog in vertebrate neural tube development. Int J Dev Biol. 2018 Mar 14;62(1-2–3):225–34.
22. Hamburger V, Hamilton HL. A series of normal stages in the development of the chick embryo. J Morphol. 1951;88:49–92.
23. Palmeirim I, Henrique D, Ish-Horowicz D, Pourquié O. Avian hairy gene expression identifies a molecular clock linked to vertebrate segmentation and somitogenesis. Cell. 1997;91:639–48.
24. Boehm B, Rautschka M, Quintana L, Raspopovic J, Jan Ž, Sharpe J. A landmark-free morphometric staging system for the mouse limb bud. Development. 2011;138:1227–34.
25. Musy M, Flaherty K, Raspopovic A J and Robert-Moreno, Richtsmeier JT, Sharpe J. A quantitative method for staging mouse embryos based on limb morphometry. Development. 2018;145:dev154856.
26. Sáenz-Ponce N, Mitgutsch C, del Pino EM. Variation in the schedules of somite and neural development in frogs. Proc Natl Acad Sci USA. 2012;109:20503–7.
27. Newgreen DF, Erickson CA. The migration of neural crest cells. Int Rev Cyt. 1986;103:89–145.
28. Bedont JL, Newman EA, Blackshaw S. Patterning, specification, and differentiation in the developing hypothalamus. WIREs Dev Biol. 2015;4(5):445–68.

29. Biran J, Tahor M, Wircer E, Levkowitz G. Role of developmental factors in hypothalamic function. Front Neuroanat. 2015;9:47.
30. Eachus H, Bright C, Cunliffe VT, Placzek M, Wood JD, Watt PJ. Disrupted-in-Schizophrenia-1 is essential for normal hypothalamic-pituitary-interrenal (HPI) axis function. Hum Mol Genet. 2017 Jun 1;26(11):1992–2005.
31. Moir L, Bochukova EG, Dumbell R, Banks G, Bains RS, Nolan PM, et al. Disruption of the homeodomain transcription factor orthopedia homeobox (Otp) is associated with obesity and anxiety. Mol Metab. 2017 Nov 1;6(11):1419–28.
32. Fu T, Towers M, Placzek MA. Fgf10+ progenitors give rise to the chick hypothalamus by rostral and caudal growth and differentiation. Development. 2017;144:3278–88.
33. Kim DW, Place E, Chinnaiya K, Manning E, Sun C, Dai W, et al. Single-cell analysis of early chick hypothalamic development reveals that hypothalamic cells are induced from prethalamic-like progenitors. Cell Rep. 2022 Jan;38(3):110251.
34. Krueger R, Beyer J, Jang W-D, Kim NW, Sokolov A, Sorger PK, et al. Facetto: Combining unsupervised and supervised learning for hierarchical phenotype analysis in multi-channel image data. IEEE Trans Vis Comput Graph. 2019;26(1):227–37.
35. Ben-Hur A, Guyon I. Detecting Stable Clusters Using Principal Component Analysis. In: Brownstein MJ, Khodursky AB, editors. Functional Genomics: Methods and Protocols [Internet]. Totowa, NJ: Humana Press; 2003 [cited 2022 Feb 7]. p. 159–82. (Methods in Molecular Biology). Available from: https://doi.org/10.1385/1-59259-364-X:159
36. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al. Going deeper with convolutions. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2015. p. 1–9.
37. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2016. p. 770–8.
38. Baltruschat IM, Nickisch H, Grass M, Knopp T, Saalbach A. Comparison of deep learning approaches for multi-label chest X-ray classification. Sci Rep. 2019;9(1):1–10.
39. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. ArXiv Prepr ArXiv14091556. 2014;
40. Hernández-García A, König P. Data augmentation instead of explicit regularization. ArXiv Prepr ArXiv180603852. 2018;
41. Margapuri V, Lavezzi G, Stewart R, Wagner D. Bombus Species Image Classification. ArXiv Prepr ArXiv200611374. 2020;

42. Cutrale F, Fraser SE, Trinh LA. Imaging, visualization, and computation in developmental biology. Annu Rev Biomed Data Sci. 2019;2:223–51.
43. Panwar H, Gupta PK, Siddiqui MK, Morales-Menendez R, Bhardwaj P, Singh V. A deep learning and grad-CAM based color visualization approach for fast detection of COVID-19 cases using chest X-ray and CT-Scan images. Chaos Solitons Fractals. 2020;140:110190.
44. Partridge M, Calvo RA. Fast dimensionality reduction and simple PCA. Intell Data Anal. 1998;2(3):203–14.
45. Ranjan S, Nayak DR, Kumar KS, Dash R, Majhi B. Hyperspectral image classification: A k-means clustering based approach. In: 4th International Conference on Advanced Computing and Communication Systems. 2017. p. 1–7.
46. Bholowalia P, Kumar A. EBK-means: A clustering technique based on elbow method and k-means in WSN. Int J Comput Appl. 2014;105(9).
47. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: machine learning in Python. J Mach Learn Res. 2011;12:2825–30.
48. Waskom M, Gelbart M, Botvinnik O, Ostblom J, Hobson P, Lukauskas S, et al. mwaskom/seaborn: v0.11.2 (August 2021) [Internet]. Zenodo; 2021 [cited 2022 Feb 7]. Available from: https://zenodo.org/record/5205191
49. Hunter JD. Matplotlib: A 2D graphics environment. Comput Sci Eng. 2007;9:90–5.
50. Bradski G. The OpenCV Library. Dr Dobbs J Softw Tools. 2000;
51. Clark A. Pillow (PIL Fork) Documentation [Internet]. readthedocs; 2015. Available from: https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf
52. Jung AB, Wada K, Crall J, Tanaka S, Graving J, Reinders C, et al. imgaug [Internet]. 2020. Available from: https://github.com/aleju/imgaug
53. Fushiki T. Estimation of prediction error by using K-fold cross-validation. Stat Comput. 2011;21(2):137–46.
54. Agarap AF. Deep learning using rectified linear units (ReLU). ArXiv Prepr ArXiv180308375. 2018;
55. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems [Internet]. 2015. Available from: https://www.tensorflow.org/
56. Goodfellow I, Bengio A Y and Courville. Deep Learning. MIT Press; 2016.
57. Zhang J, Mitliagkas I. YellowFin and the Art of Momentum Tuning. Proc Mach Learn Syst. 2019 Apr 15;1:289–308.

58. Chollet F, others. Keras [Internet]. 2015. Available from: https://keras.io
59. Gabas A, Corona E, Alenyà G, Torras C. Robot-aided cloth classification using depth information and CNNs. In: International Conference on Articulated Motion and Deformable Objects. Springer; 2016. p. 16–23.
60. Srivastava N. Improving neural networks with dropout. Univ Tor. 2013;182(566):7.
61. Yasuhiro K. tf-keras-vis [Internet]. GitHub repository. GitHub; 2021. Available from: https://github.com/keisen/tf-keras-vis
62. Smilkov D, Thorat N, Kim B, Viégas F, Wattenberg M. Smoothgrad: removing noise by adding noise. ArXiv Prepr ArXiv170603825. 2017;
63. LeNail A. NN-SVG: publication-ready neural network architecture schematics. J Open Source Softw. 2019;4(33):747.

FIGURES

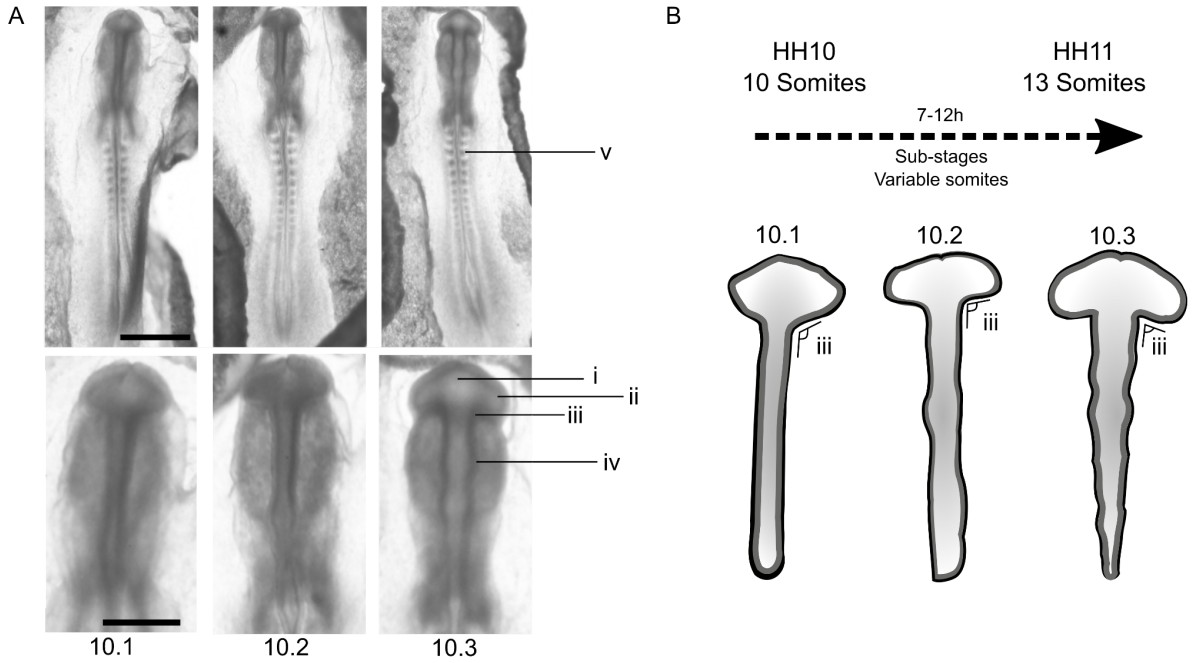


Fig 1. Sub-staging the HH10 chick embryo brain based on expert visual inspection of morphology. (A) Example brightfield microscopy images of stage 10 sub-stages (top panels), with close-ups of the developing brain (bottom panels). These images were used as the original reference point for ground truth labelling, and were not used as part of the DNN classifier training dataset. (i-iii) show key anatomical markers used for sub-stage labelling. (i): prosencephalon, (i): optic vesicle, (iii): angle of prosencephalon and prosencephalic neck. (iv, v): show other anatomical features relevant to this study: (iv) somites; (v) neuromere. Scale bars: top 1000μm, bottom 500μm. (B) Schematic indicating morphological profiles of sub-stages.

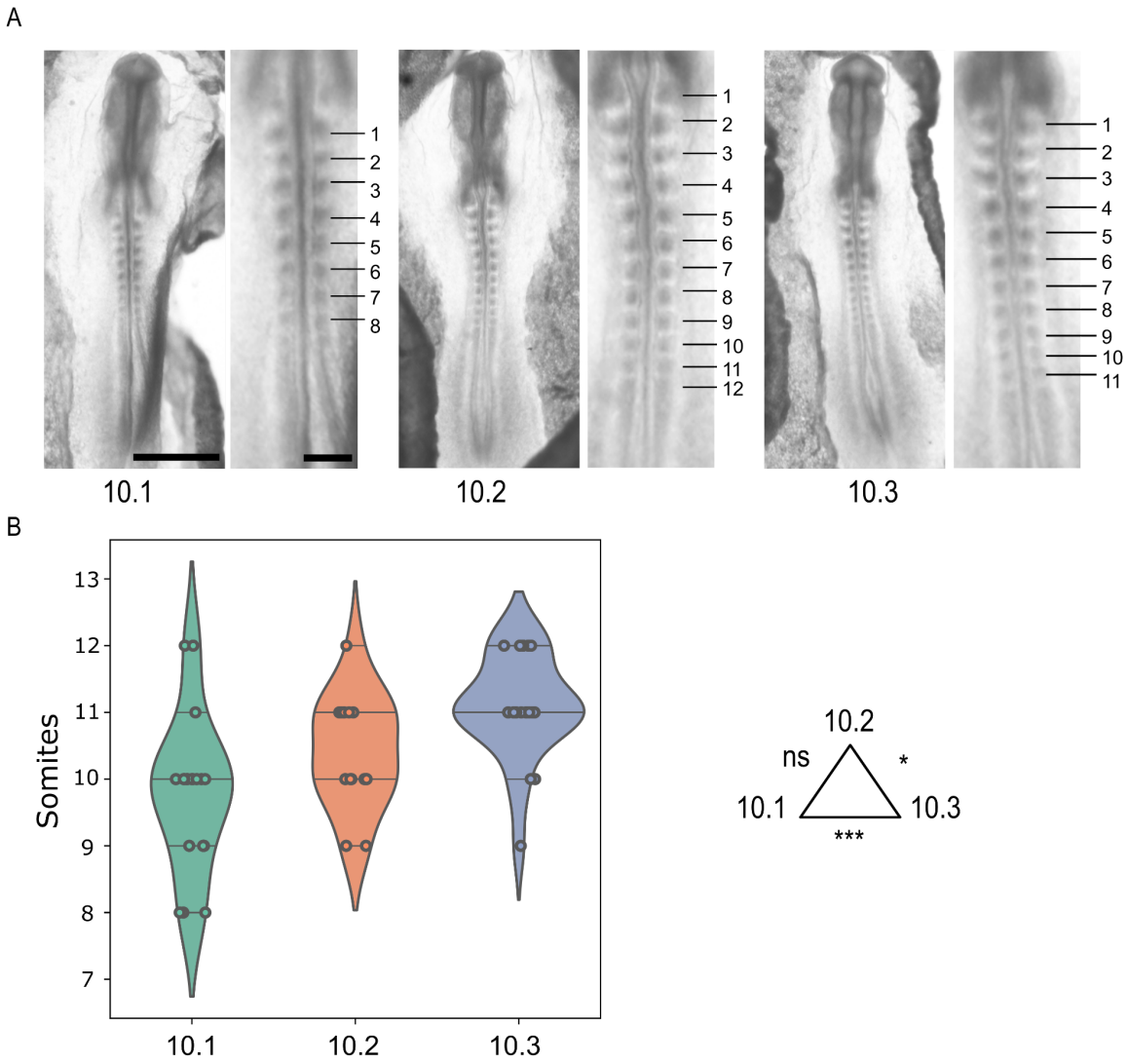


Fig 2. The number of somites at the 10.1, 10.2, and 10.3 sub-stages does not account for the variation seen in HH10 brain morphology. (A) Manual quantification of somite number for the reference images shown in **Fig 1A**. Right hand panels show close-ups of somitic regions. Scale bars: left 1000μm, right 250μm. (B) Violin plot of somite count across sub-stages 10.1, 10.2 and 10.3. (n= 18 10.1, 14 10.2, and 21 10.3 embryos respectively: note the data used for these quantifications were not included as training data). There are statistically significant differences in somite numbers between 10.1 and 10.3, and 10.2 and 10.3, but not 10.1 and 10.2. Despite these differences, each sub-stage can have 9, 10, 11, or 12 somites.

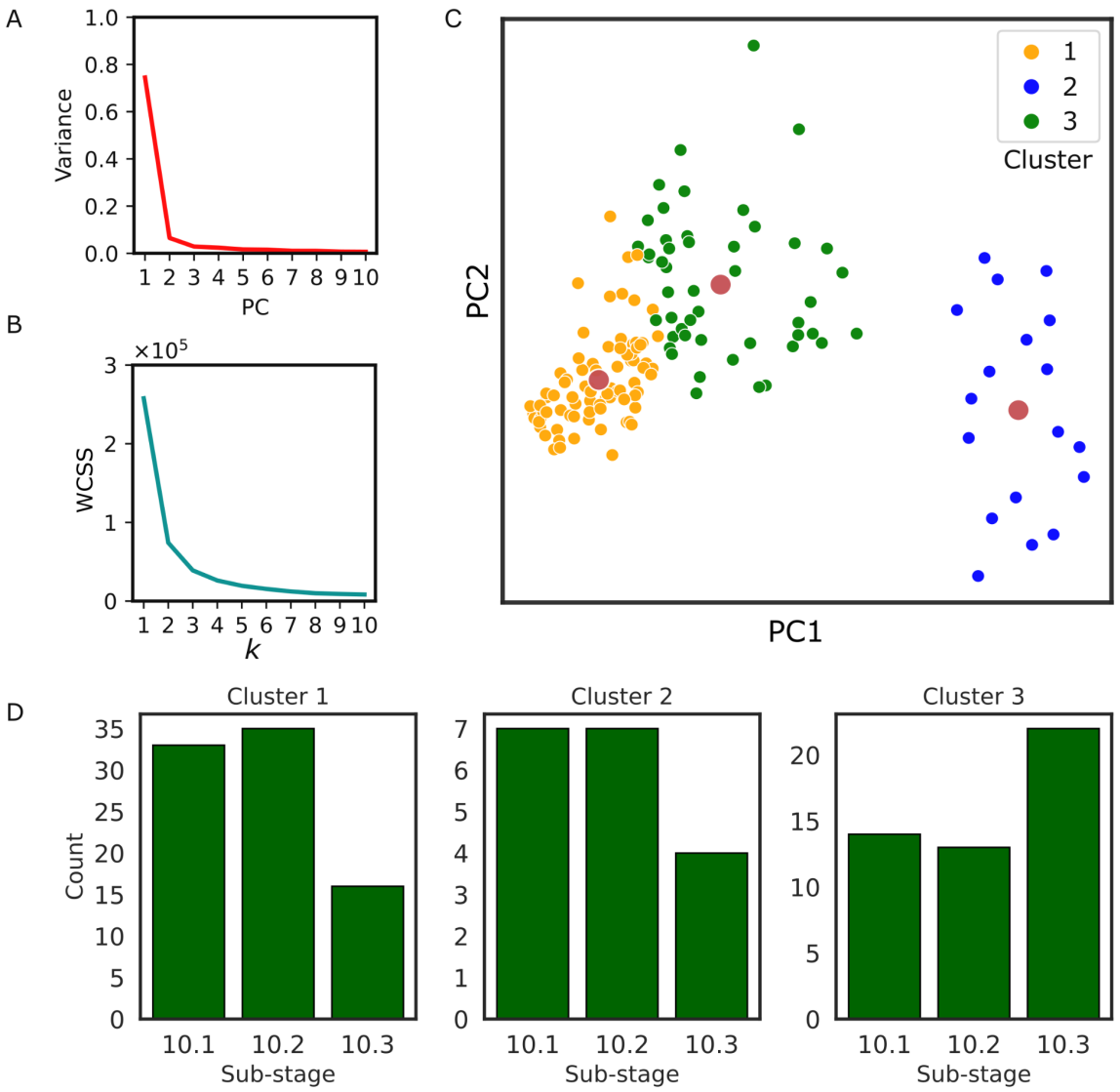


Fig 3. *k*-means clustering of the image dataset. (A) Scree plot for principal component analysis (PCA) with increasing numbers of PCs. The elbow point is at 2 PCs, which explain 81% of the variance in the dataset. (B) The elbow method for determining the optimal number for *k*. Here, the within-cluster sum of squares (WCSS) score is calculated for *k*-means clustering on increasing values of *k*. The optimal number of *k* is determined by the inflection point of the graph, which occurs around *k* = 3. (C) Scatter plot of the *k*-means clustered dataset, with *k* centroids (red circles). (D) Number of sub-stages present in each cluster. Despite unsupervised methods identifying three groups of images, the number of embryos in each cluster does not match the number of embryos in each labelled sub-stage of the training data.

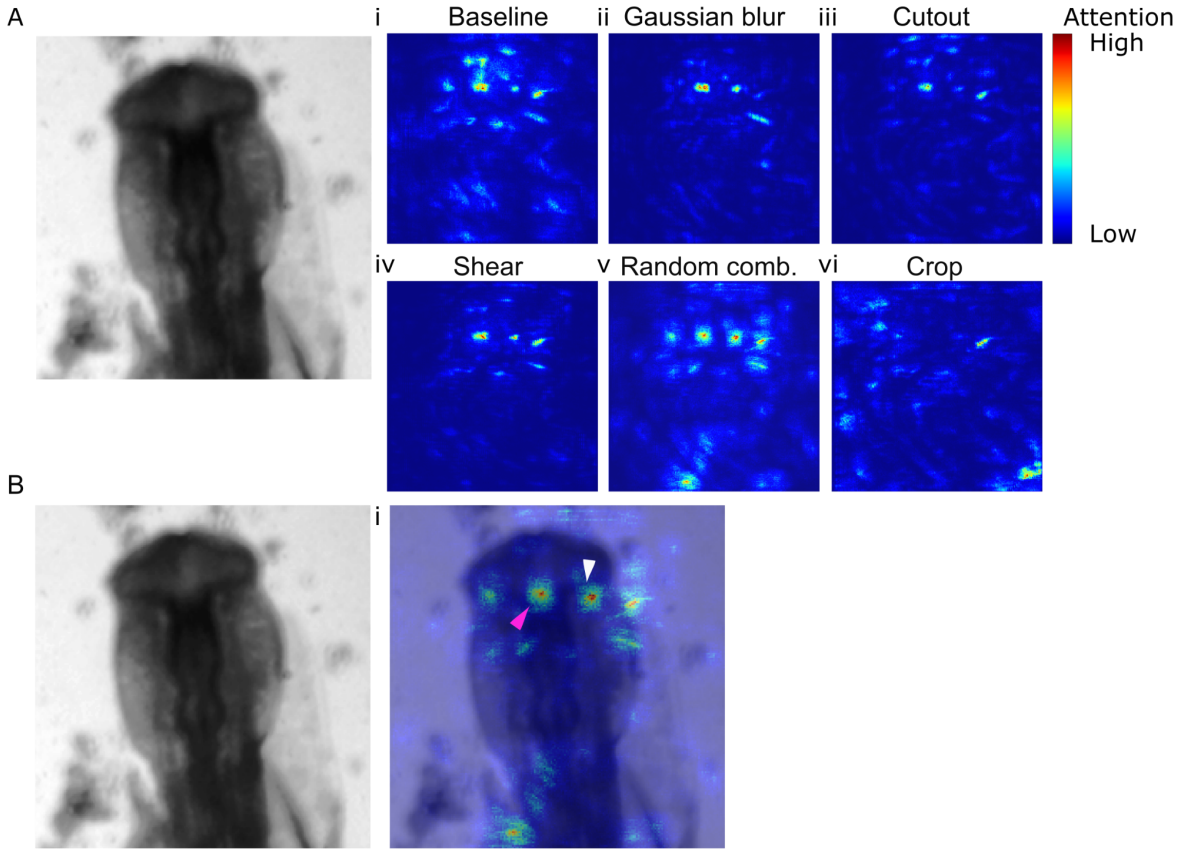


Fig 4. Saliency maps of a 10.2 substage identify focal differences between models trained on different augmentations. (A) A test 10.2 image not used in training/validation of any DCNN. (i-vi) Saliency maps generated from training a DCNN on the baseline rotationally augmented data (i), with a further augmentation (ii-vi, listed above each image). Baseline (i) leads to a large amount of attention spread over the prosencephalon. Gaussian blur (ii) cutout (iii), and shear (iv) increase focus to a few focal points on the prosencephalon. The maps generated by the augmentations i-v contrast with crop (vi). This performed the worst of all our augmentation regimes, and has very little focus on the prosencephalon. (B) (i) The saliency map generated by the combined regime as shown in (Av), overlaid with the test 10.2 image. The combined regime increases the focus on points introduced by the various augmentations, and the focus is highest on the region of the embryo (i, magenta arrow) central to the prosencephalic neck (white arrow), also a region of high focus. The combined regimes of various augmentations increase the focal attention. Highest focal points are the centre of the prosencephalic neck (magenta arrow) and the junction of the prosencephalon-prosencephalic neck (white arrow).

TABLES

Aug	Fold											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Avg.**	**SD**
1	40.7	41.5	43.3	46.7	42.4	46.1	47.0	46.7	46.5	40.4	44.1	**2.7**
1 + 2	**41.0**	40.0	53.3	49.1	**44.9**	38.1	46.8	46.9	47.5	39.4	44.7	4.9
1 + 3	35.6	**50.7**	51.3	**49.6**	43.3	47.6	46.2	**47.1**	32.9	**49.1**	45.3	6.3
1 + 4	40.0	35.4	54.4	40.9	40.9	46.6	46.7	46.7	**48.7**	42.6	44.3	5.4
1 + 5	40.1	41.9	53.8	45.4	36.2	47.5	46.7	46.7	46.8	40.1	44.5	5.0
1 + 2,4,5 RC	38.0	42.2	**60.5**	38.7	40.9	**52.8**	**48.6**	46.0	47.1	37.8	**45.4**	7.4
Fold Avg.	39.2	42.0	52.8	45.1	41.4	46.5	47.0	46.7	45.1	41.6		

Table 1. Augmentation exploration of the dataset using InceptionV3. We explored the data-space using *k*-fold cross validation, the individual fold validation accuracies that each network achieved are shown in columns 1-10, and the averages and standard deviation of these accuracies is shown in the rightmost columns. As a baseline processing step, all images were rotated 15 times, at equally spaced degrees. We then tested augmentations on top of this baseline, before a final test whereby each image was randomly augmented with augmentations 1, 2, 3, 4, or 5. The augmentations (Aug) are as follows: (1) rotation (baseline); (2) shear; (3) crop; (4) Gaussian blur; (5) cutout; (RC) random combination of rotation + cutout, or shear, or blur. Highest validation accuracies for each fold, highest average for each augmentation (Avg.), and lowest standard deviation (SD) are shown in bold.

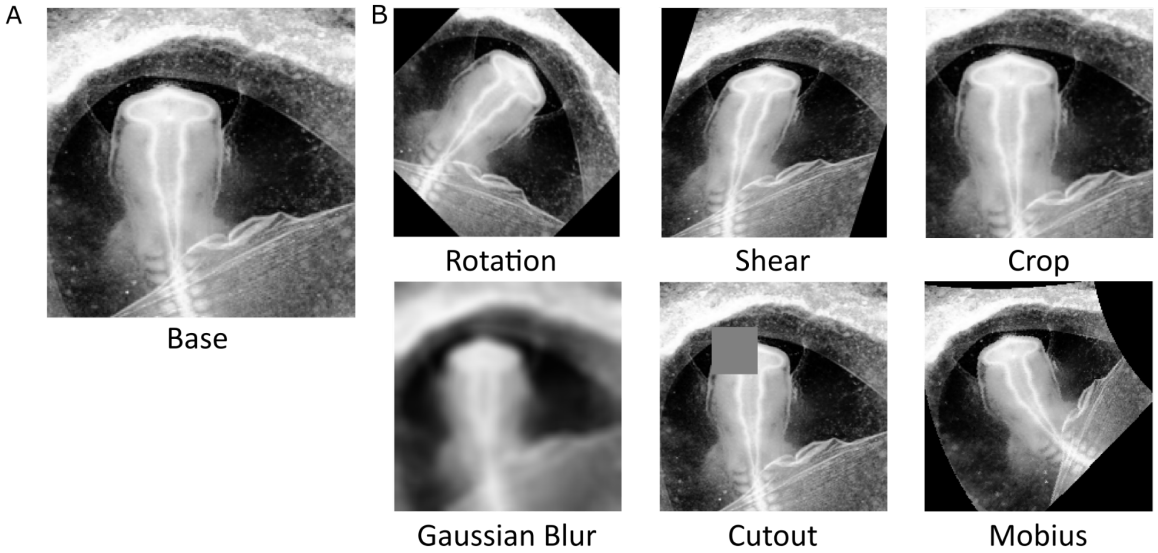
Aug	Fold											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Avg.**	**SD**
1	40.0	71.1	73.5	54.1	69.4	66.9	64.1	70.9	65.9	63.7	64.0	10.0
1 + 2	65.4	40.1	67.0	51.0	69.8	67.1	63.4	48.7	67.9	56.2	59.9	11.7
1 + 3	44.2	55.6	60.9	52.4	51.8	46.7	46.7	50.0	45.8	46.9	50.1	**5.2**
1 + 4	**69.0**	**74.5**	**75.9**	51.1	**70.6**	69.2	**70.6**	70.1	**71.6**	**69.4**	**69.2**	6.8
1 + 5	45.7	40.1	56.9	52.0	44.3	**70.4**	69.8	49.9	71.3	48.2	54.9	11.7
1 + 2,4,5 RC	63.1	37.2	68.7	48.1	67.3	62.7	52.0	**71.0**	68.4	55.3	59.4	11.0
Fold Avg.	54.6	53.1	67.2	51.5	62.2	63.8	61.1	60.1	65.2	56.6		

Table 2. Augmentation exploration of the dataset using ResNet50. We explored the data-space using *k*-fold cross validation, the individual fold validation accuracies that each network achieved are shown in columns 1-10, and the averages and standard deviation of these accuracies is shown in the rightmost columns. As a baseline processing step, all images were rotated 15 times, at equally spaced degrees. We then tested augmentations on top of this baseline, before a final test whereby each image was randomly augmented with augmentations 1, 2, 3, 4, or 5. The augmentations (Aug) are as follows: (1) rotation (baseline); (2) shear; (3) crop; (4) Gaussian blur; (5) cutout; (RC) random combination of rotation + cutout, or shear, or blur. Highest validation accuracies for each fold, highest average for each augmentation (Avg.), and lowest standard deviation (SD) are shown in bold.

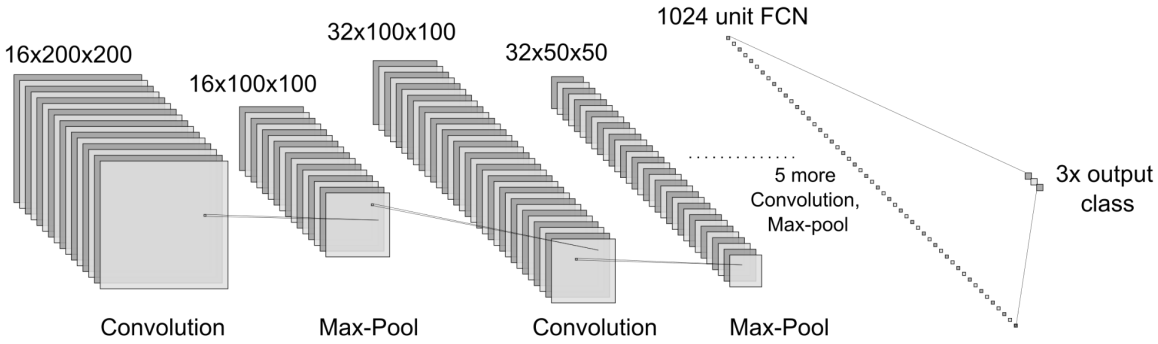
Aug	Fold											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Avg.**	**SD**
1	69.3	68.1	79.3	78.9	72.6	84.8	74.6	68.0	90.9	75.2	76.2	7.5
1 Sharpened	78.7	80.9	77.0	70.2	80.2	80.7	70.7	76.7	90.0	76.9	78.2	5.6
1 + 2	68.9	75.5	78.8	79.8	75.1	84.5	78.8	69.1	90.3	83.1	78.4	6.7
1 + 3	45.3	46.2	47.6	51.3	46.0	51.3	44.7	45.8	52.7	40.0	47.1	**3.8**
1 + 4	81.0	73.8	80.6	**79.1**	**81.7**	**86.9**	**79.0**	71.9	88.9	81.9	**80.5**	**5.1***
1 + 5	73.6	76.6	79.9	76.7	76.9	86.1	78.0	68.1	**91.7**	83.3	79.1	6.6
1 + 4,5 RC	**86.3**	**81.0**	78.2	74.2	77.6	82.2	73.2	**77.6**	89.0	83.2	80.3	**5.1***
1 + 2,4,5 RC	84.6	75.5	**82.1**	70.1	81.2	83.1	75.9	77.1	90.9	**84.2**	**80.5**	5.9
Fold Avg.	70.5	69.2	74.7	73.7	72.3	79.5	71.8	66.7	83.2	74.6		

Table 3. Augmentation exploration of the dataset using the bespoke model. We explored the data-space using *k*-fold cross validation, the individual fold validation accuracies that each network achieved are shown in columns 1-10, and the averages and standard deviation of these accuracies is shown in the rightmost columns. As a baseline processing step, all images were rotated 15 times, at equally spaced degrees. We then tested augmentations on top of this baseline, before a final test whereby each image was randomly augmented with augmentations 1, 2, 3, 4, or 5. The augmentations (Aug) are as follows: (1) rotation (baseline); (1 Sharpened) rotation, with a sharpening filter as an additional pre-processing step; (2) shear; (3) crop; (4) Gaussian blur; (5) cutout; (RC) random combination of rotation + cutout, or shear, or blur. The maximum accuracy for each augmentation is shown in bold. Maximum validation accuracies for each fold, highest average for each augmentation (Avg.), and lowest standard deviation (SD) are shown in bold.

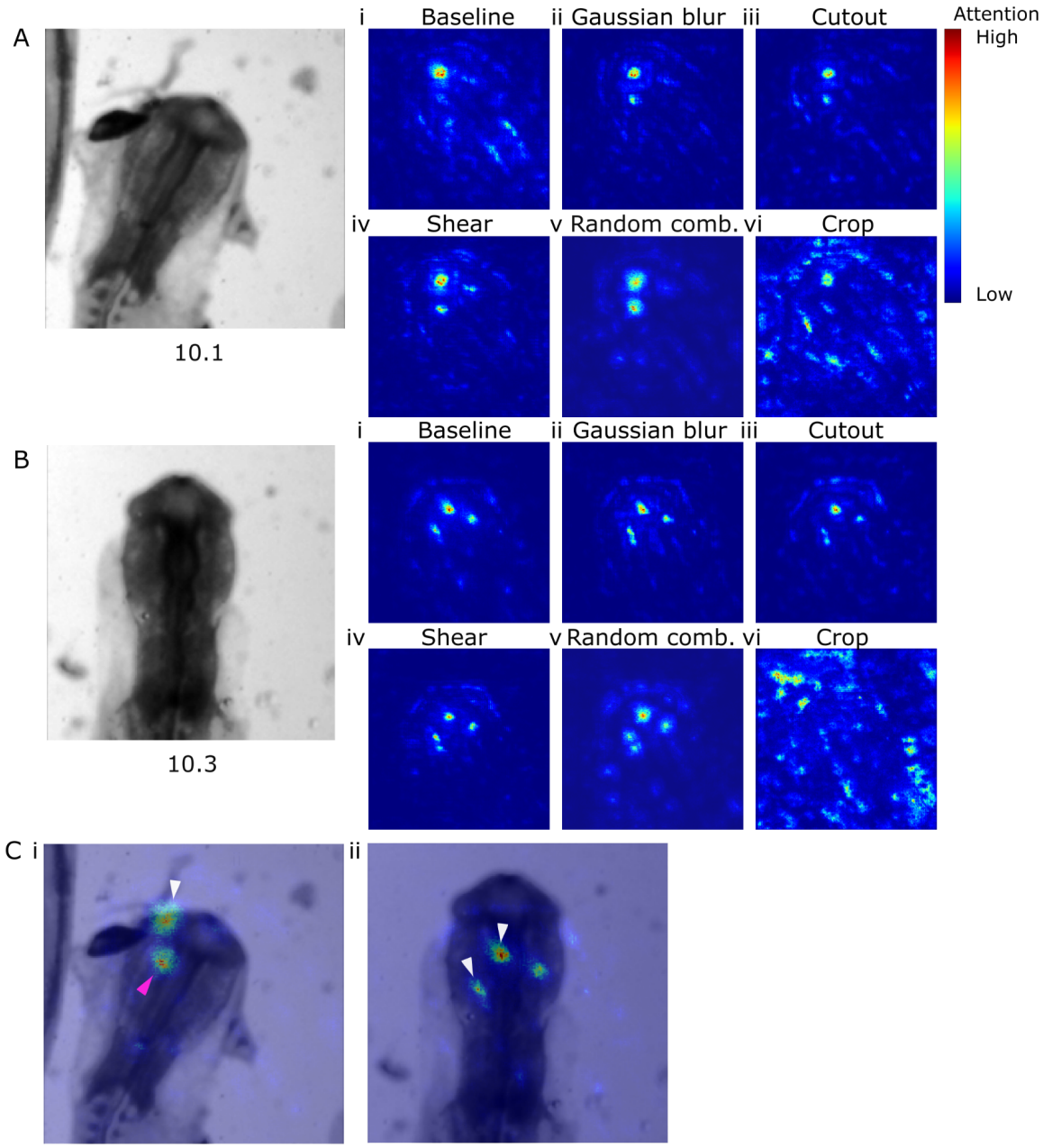
SUPPLEMENTARY FIGURES



S1 Fig. Applying different augmentations to an image of a 10.2 sub-stage embryo. (A) An example image of a 10.2 embryo from the dataset (base), with (B) the augmentation regimes tested for classifier training: geometric (e.g. rotation), photometric (e.g. contrast), and complex (e.g. cutout) augmentations.



S2 Fig. Image classifier network architecture. After the input layer, the image was processed following the VGG block style, i.e. through increasingly wide convolution and max pooling layer pairs (39). At each 'block' we used a single convolutional layer which retains resolution through padding. Following this, a max pooling layer is used as a method of spatial downsampling. Between each convolutional layer and max pooling layer there is a non-linear ReLU activation function and a dropout layer. For simplicity, only 2 of the blocks are shown. After the final max pooling layer, the data was passed through a wide fully connected layer. Lastly, the classification is made in the output layer. Diagram generated using NN-SVG (63).



S3 Fig. Saliency maps of 10.1 and 10.3 substages identify focal differences between models trained on different augmentations. (A-B) Test images of 10.1 and 10.3 embryos. (i-vi) Saliency maps generated from (A-B) and assessed through models trained on different augmentation regimes, following our architecture (from **Table 3**), trained on the baseline rotationally augmented data, with a further augmentation (listed above each image). Random comb. refers to a model trained on our baseline and a random combination of Gaussian blur, cutout, and shear augmentations. The random combination results in attention which is discrete and focused e.g. (Av), (Bv). Conversely, models trained on

baseline & crop show scattered attention, where the attention map shows high attention pixels scattered throughout the image can be seen (Avi,Bvi). A scattered attention map suggests that the model is focusing on various image features unrelated to the subject. (C) Saliency maps as in (A,B v) overlaid with the test image. Here, for 10.1, the model focused most on the angle of the prosencephalic neck (Ci magenta arrow) and the anterior prosencephalon (Ci white arrow). For 10.3 the focus was shifted to more posterior regions of the embryo, including the neuromeres (Cii, white arrows).

SUPPLEMENTARY TABLES

S1 Table. Testing of Möbius transformations as data augmentations. 1 + Möbius: the dataset is augmented with our baseline & Möbius transformations. 1 + M:G (10% chance). The dataset is augmented with Gaussian blur with a 10% chance of a Möbius transformation per image.

Aug	Fold											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Avg.**	**SD**
1 + Möbius	55.7	57.6	47.4	49.1	49.4	41.3	38.1	57.4	43.9	66.1	50.6	8.6
1 + M:G (10% chance)	76.6	79.4	67.4	66.4	79.8	76.9	70.8	77.6	84.6	79.0	75.9	5.8

S2 Table. Data augmentation regime exploration with hold-out cross-validation. The networks were initialised and trained with reshuffled data 10 times for each augmentation. Rotation achieves the highest accuracy at 91.6%, however cutout reaches 91.5%, and is the most reliable, with the highest average and lowest standard deviation. The optimal score for each metric is shown in bold.

	Avg. (%)	SD	Min (%)	Max (%)
Rotation	80.3	6.7	64.7	**91.6**
Crop	74.8	8.4	59.9	82.3
Shear	78.4	4.8	68.1	84.8
Gaussian blur	80.3	4.9	71.9	89.3
Cutout	**83.6**	**4.6**	**76.1**	91.5
Salt pepper	74.4	7.4	61.9	87.6
Gamma correction	78.7	6.5	67.2	87.7
Crop and translation	53.5	8.9	39.2	74.9
Background removal	73.1	6.2	65.0	87.2

S3 Table. Optimal hyperparameters for our baseline, determined by Bayesian optimisation. We tested these hyperparameters with the following ranges: Activation function: ReLU-Sigmoid, Batch size: 16-128, Optimiser: Adam, Adadelta, Adamax, Adagrad, SGD, RMSprop, Layer dropout: 0.05-0.25%. Final layer dropout: 0.3-0.8%. λ: 10^{-3} - 10^{-6}, Learning rate: 10^{-1} - 10^{-6}, selecting the value/category which was used most by the optimisation algorithm.

Hyperparameter	Value/category
Activation function	ReLU
Batch size	32
Layer dropout	20%
Final layer dropout	50%
L_2 regularisation λ	10^{-4}
Optimiser	Adam
Learning rate	10^{-4}
Average validation accuracy	64.1%
Min, Max validation accuracy	43.6%, 81.5%