Detecting Operons in Bacterial Genomes via Visual Representation Learning

Rida Assaf[1,*], Fangfang Xia[3,4], and Rick Stevens[2,3]

[1]University of Chicago, Department of Computer Science, Chicago, 60637, USA
[2]University of Chicago, The University of Chicago Consortium for Advanced Science and Engineering, Chicago, 60637, USA
[3]Argonne National Laboratory, Computing Environment and Life Sciences Division, Lemont, 60439, USA
[4]Argonne National Laboratory, Data Science and Learning Division, Lemont, 60439, USA
[*]rida@uchicago.edu

ABSTRACT

Genes in prokaryotic genomes assemble in clusters, forming transcription units called operons. Detecting operons plays a critical role in inferring gene functionality and detecting regulatory networks. Several tools have been proposed to detect operons computationally. Many of these tools are no longer maintained and have not been evaluated with recent data. Inspired by how human experts annotate operons by visually inspecting gene neighborhoods across close relatives, we propose a novel deep learning based method (named Operon Hunter), that uses visual representations of genomic fragments to make operon predictions. The visual representations are engineered to capture the most prominent operon features discussed in the literature, such as inter-genic distance, strand direction, gene size, functional relatedness, and gene neighborhood conservation. Our method uses a pre-tained network via transfer learning to leverage the power of deep neural networks trained on big datasets. The network is re-trained on a limited dataset of extensively validated and experimentally verified operons.
We report the highest accuracy in the literature that we know of when testing on the standard datasets that are reported in various operon prediction studies. Our validation dataset consists of two well-studied genomes with validated operons: E. coli and B. subtilis. Operon Hunter achieves a 0.92 F1 score, matched with a similar true negative rate, outperforming the previously reported state-of-the-art methods. Additionally, in contrast to most available tools that make isolate gene-pair predictions, our method demonstrates a clear advantage when it comes to detecting full operons and their boundaries.

Introduction

In prokaryotes, contiguous genes are often arranged into operons. These genes share a common promoter and terminator[1], and are usually metabolically or functionally related. The genes constituting an operon are grouped together on the same strand and are transcribed in the same polycistronic messenger RNA[2–12]. We reserve the definition of operons as clusters including at least two genes, distinguishing them from Transcription Units, which may include one or more genes[2]. It is estimated that more than 50% of bacterial genes form operons[13]. Since operons are often formed by genes that are related functionally or metabolically, predicting operons could help us predict higher levels of gene organization and regulatory networks[2,3,8,9,11,12,14]. Such predictions could help annotate gene functions[15], aid in drug development[16], and the work on antibiotic resistance[17].

Different methods use different operon features to make their predictions. Some methods use Hidden Markov Models (HMMs) to find shared promoters and terminators[18–20]. Other methods rely on gene conservation information[21], while others leverage functional relatedness between the genes[2,22]. The most prominent features used in operon prediction are transcription direction and inter-genic distances, as reported in the literature[2,8,13,23–32]. Gene conservation is another important feature, since adjacent genes that are co-transcribed are likely to be conserved across multiple genomes[11,33]. Different machine learning (ML) technologies are used to predict operons, such as neural networks[23,24], support vector machines[25], and decision tree-based classifiers[26]. Other tools utilize Bayesian probabilities[27–29], genetic algorithms[30], and graph-theoretic techniques[22,28]. The majority of available tools perform their predictions on consecutive gene pairs. To get full operons predictions, a tool generates a prediction for every available gene pair in a genome, then combines the contiguous pairs that are labeled as operonic into full operons.[2,4]. Two genomes are extensively used to verify the majority of the published tools' results: E. coli and B. subtilis. These are two well-studied studied genomes with rich datasets of experimentally validated operons, which makes them desirable candidates for benchmarking.[4].

We focus our attention on two machine learning based tools. The first is reported to have the highest accuracy among operon prediction methods[7]. It is based on an artificial neural network that uses inter-genic distance and protein functional relationships[8]. To infer functional relatedness, the method uses the scores reported in the STRING Database[34]. The STRING database captures functional relatedness through scores generated using information about gene neighborhood, fusion, co-occurrence, co-expression, protein-protein interactions, in addition to information extracted by automatic literature mining. The predictions made by this method were compiled into what is called the Prokaryotic Operon Database (ProOpDB)[9], and released later as a web service called Operon Mapper[10]. For simplicity, we will refer to this method as ProOpDB, given that it was our resource for the predictions over the model organisms we used for the cross-tool comparison. The second tool is called the Database of Prokaryotic Operons (Door)[3] and was ranked as the second best operon predictor after ProOpDB by the same study[7]. It was also ranked as the best operon predictor among 14 tools by another independent study[35]. Door's algorithm uses a combination of a non-linear (decision-tree based) classifier and a linear (logistic function-based classifier) depending on the number of experimentally validated operons available for the genomes used in the training process. Among the features Door uses to perform its predictions for adjacent gene pairs are: the distance between the two genes, the presence of a specific DNA motif in the genomic region separating them, the ratio of the genes' lengths, the genes' functional similarity (determined using Gene Ontology (GO)), and the level of conservation of the genes' neighborhood.

Before delving into the details of our method, we point out some of the challenges that undermine operon prediction in general. One limitation that faces predictors that rely on features such as gene conservation or functional assignment is the requirement to have such information about all the genes in a genome. So while such predictors might perform well on gene pairs that include the necessary features, their performance might drop considerably when making predictions over the entire genome[2]. Moreover, even though most methods validate their results by comparing their predictions over experimentally verified operons, the fact that the experimentally verified datasets are only available for a small subset of the sequenced genomes and that the datasets used vary between studies poses extra challenges making the comparison between the available tools non-straightforward. Brouwer et al tried to compare several methods using a uniform dataset and noticed a significant gap between the measures achieved and those reported in their original papers. The drop in performance was even higher when considering full operon predictions rather than separate gene pair predictions[7,35]. Finally, Some methods include the testing dataset as part of the training dataset, which leads to a reported accuracy that is significantly higher than what would be otherwise, given that the flow of information taking place in the training process would not be easily and readily transferable to novel genomes used as a testing dataset[8]. In the next section, we describe our method that uses a pre-trained neural network and visual representations of genomic regions to alleviate some of these challenges.

Methods

Using graphical representations makes it possible to leverage the powerful ML technologies that have become the state of art in solving computer vision problems. Algorithms based on Deep Neural Networks have proven to be superior in competitions such as ImageNet Large Scale Visual Recognition Challenge (ILSVRS)[36]. Deep learning is the process of training deep neural networks (i.e neural networks with many hidden layers). The depth of these networks allows them to learn complex patterns and higher order relationships, at the cost of being more computationally expensive and requiring more data to work effectively. Improvements in such algorithms have been translated to improvements in a myriad of domains reliant on computer vision tasks[37].

In our approach, the images were generated by PATRIC (The Pathosystems Resource Integration Center), which is a bacterial Bioinformatics resource center that we are part of (https://www.patricbrc.org)[38]. More specifically, one of the services PATRIC provides is the compare region viewer service, where a query genome is aligned against a set of other related genomes anchored at a specific focus gene. The service starts with finding other genes that are of the same family as the query gene, and then aligning their flanking regions accordingly. Such graphical representations are appealing as they help users visualize the genomic areas of interest. We replicated the service by implementing an offline version forked from the production UI, which is necessary for computational efficiency and for consistency in the face of any future UI changes.

Each row of arrows in the generated image represents a region in a genome, with the query genome being the top row. Each arrow represents a single gene, scaled to reflect its size relative to its region, in addition to the gene's strand directionality. The distances between the genes are also scaled on each row relative to the gene's region. Each image consists of three regions. The central region makes up two thirds of the image, and represents the query gene pair. Genes that fall before the query gene pair are represented to the left of the central region, and those that fall after the query gene pair are represented to the right of the central region. Dividing the image implicitly into three regions highlights the area of most interest by zooming in on the query gene pair, while preserving the relevant conservation information of the flanking genomic fragments on the sides. Colors represent gene functionality. The blue and red arrows are reserved to represent the query gene pair and the rest of the genes that belong to the same families. In general, genes share the same color if they belong to the same family, or are colored black after a certain color distribution threshold. The families used in the coloring process are PATRIC's Global Pattyfams that

are generated by mapping signature k-mers to protein functionality, using non-redundant protein databases built per genus before being transferred across genera[39]. Finally, the image's alpha channel is set to be equal to the STRING score of the query gene pair. If no score exists, the default alpha channel is set to a minimum of 0.1. The generated images capture most of the prominent features mentioned earlier, such as gene conservation, functionality, strand direction, size, and inter-genic distance. Figure 1 shows an example of how the generated images look like.

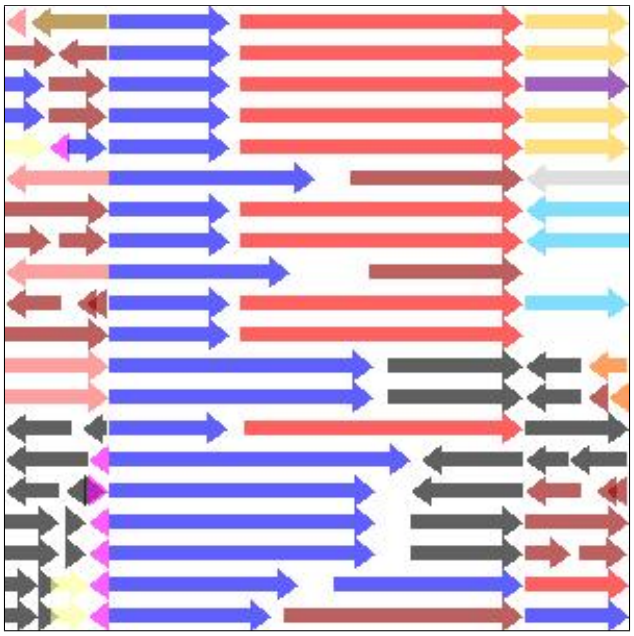


Figure 1. Example of an image generated by our service to be fed as input to the neural network. Each arrow represents a single gene. Each row captures the area of interest in a genome. The query genome is the top row. The rest of the rows are genomes selected by evolutionary distance. The query gene pair are colored blue and red. Genes share the same color if they belong to the same family and that family. The query gene pair are centered in the middle, occupying 2/3 of the image's width. The rest of the flanking region is represented correspondingly to the left/right of the center region. The alpha channel of the image is the STRING score of the query gene pair.

Images offer a natural way to compare genes (horizontally) and clusters across genomes (vertically) with 2D convolution. The fact that the compare region view sorts genomes by evolutionary distance allows the neural network to exploit locality and place more emphasis on close genomes via incremental pooling. An additional benefit of working with the image modality is to be able to leverage the state-of-the-art deep learning models, many of which were first developed in vision tasks and perfected over years of iterations. In our previous work[40], we demonstrated a method that uses images representing genomic data to identify genomic islands. Moreover, Google researchers have used spectrogram (instead of text) in direct speech translation[41] and DNA sequence pile-up graphs (instead of alignment data) in genetic variant calling[42]. In both cases, the image-based models outperformed their respective previous state-of-the-art method based on traditional domain features. Further, the low-level visual feature patterns learned in pre-trained image models have been demonstrated to transfer to distant learning tasks on non-image data in several preliminary studies ranging from environmental sound classification to cancer gene expression typing[43]. Much like feature engineering methods, casting tabular data to images encodes information in a way more amenable to learning without explicitly adding information. It can also be easily integrated with other data modalities in the latent vector representation to prevent information loss. We hypothesize this emerging trend of representing data with images will continue until model tuning and large-scale pre-training in scientific domains start to catch up with those in computer vision.

We used the "Known Operons" section of the Operon DataBase (ODB)[46] to construct the training dataset. The Known Operons section contains a list of operons that are experimentally verified, which we used to label the generated images. Out of the entire dataset, the six genomes (Table 1) with the most number of labeled operons were selected. These genomes have significantly more operons than the other genomes listed in this section of the database. For each of the selected genomes, our program produces an image for every consecutive pair of genes that are part of a validated operon. To generate the images, each of these genomes is aligned with the set of reference+representative genomes found on PATRIC. For every aligned gene, an image is generated to capture the surrounding 5 kilo base pairs (Kbp) flanking region. The resulting dataset consisted of 4,306

images of operonic gene pairs. To generate the dataset representing the non-operonic gene pairs, we used the standard approach reported by ProOpDB[9] as follows: Genes that are at the boundaries of known operons are labeleled along with the respective upstream or downstream gene that is not part of that operon as a non-operonic gene pair. We skipped single-gene operons from the training dataset. A balanced dataset was then curated from the total set of images created.

Table 1. Breakdown of the training dataset: The genome names and the corresponding number of gene pairs used. Operon pairs were harvested from the Known Operons section of ODB.

Genome name	Operon pairs	non-Operon pairs
Escherichia coli	1,443	1,322
Listeria monocytogenes	806	780
Legionella pneumophila	611	791
Corynebacterium glutamicum	525	396
Photobacterium profundum	447	544
Bacillus subtilis	474	457
Total	4,306	4,290

The relatively small size of the dataset combined with the depth of the network pose the risk of over-fitting. One way to get around this issue is by using a technique referred to as transfer learning[44]. In transfer learning, a model does not have to be trained from scratch. Instead, a model that has been previously trained on a related task is retrained on the new task. The newly retrained model should then be able to transfer its existing knowledge and apply it to the new task. This approach makes it possible to be able to reuse models that have been trained on huge amounts of data, by adding the necessary adjustments to make them available to work with more limited datasets. This adds a further advantage to representing the data visually. To train and test our model, we used the FastAI[45] platform. After testing the performance of the different available pre-trained models on the operon dataset, the best performance was observed when using the ResNet18 model. All available models were previously trained on Imagenet, which is a database that contains more than a million images belonging to more than a thousands categories[36]. Thus, a model that was previously trained on ImageNet is already good at feature extraction and visual recognition. To make the model compatible with the new task, the top layer of the network is retrained on the operon dataset, while the rest of the network is usually left intact. This is more powerful than starting with a deep network with random weights. Another technique that is commonly used against over-fitting is data augmentation, whereby the training dataset is enriched with new examples by applying certain transforms on the existing images. These transforms include flips(horizontal and vertical), zooming effects, warping, rotation, and lighting changes. Using FastAI's toolkit, we augmented our training dataset by allowing only one transform, the horizontal flip, to be applied. Given the deliberate feature engineering processes taken to create the input images, we believe that horizontal flips could safely be applied without altering the true nature of the images. This would keep the key information intact and in place(e.g. by keeping the query genome as the top row).

All tools perform their analysis and make their predictions on pairs of consecutive genes. We refer to gene pairs that are part of an operon as operonic, and gene pairs where one is a boundary of an operon, or that include a verified operon consisting of a single gene, to be non-operonic.

We resort to two extensively studied genomes with experimentally verified operons: E. coli and B. subtilis. These genomes are the standard for verifying operon prediction results in the literature. We limit our validation by using only the experimentally validated operons found in these genomes. We compare the predictions made by Operon Hunter to those made by ProOpDB and Door, the tools with state of the art accuracies as reported by independent studies[3,7,9]. To build the testing dataset (Table 2), we used the operons published in RegulonDB[47] and DBTBS[48]. RegulonDB is a database containing verified operons found in E. coli, and DBTBS is a database that contains verified operons found in B. subtilis. We cross check the operons found in these databases, with the ones published in the Known Operons section of ODB. We selected the operons that are reported in both: The database corresponding to its organism (RegulonDB for E. coli, DBTBS for B. subtilis), and ODB. The resulting dataset consists of 730 operonic gene pairs. We used the same approach mentioned earlier to construct a non-operonic dataset. Namely, the boundaries of the operons were selected along with their neighboring genes as non-operonic pairs. To construct the non-operonic datasets, it was enough for an operon to be published in any of the mentioned databases to be considered, resulting in a slightly larger non-operon dataset. Using operons that were experimentally verified and published in multiple independent databases adds confidence to the assigned label.

Some tools train on the same organisms they report their predictions on. In our experiments, we exclude the testing dataset from the training dataset. For example, when evaluating the model's predictions on the E. coli dataset, it is trained on all the images representing all the genomes except E. coli, then tested on the images representing the the E. coli genome. Thus, none of the genomes used in the training process belong to the same organism as the genome used for testing our method's performance.

Table 2. Breakdown of the testing dataset: The genome names and the corresponding number of gene pairs used. Operon pairs were scoured from RegulonDB (for E. coli) and DBTBS (for B. subtilis) and matched with the Known Operons section of ODB.

Genome	Operon pairs	non-Operon pairs
Escherichia coli str. K-12 substr. MG1655	361	461
Bacillus subtilis subsp. subtilis str. 168	369	302
Total	730	763

Results

Evaluating Positive-Class Predictions

We report the sensitivity (true positive rate), precision, and the corresponding F1 measure to capture the tools' performance over the positive class in Table 3, Table 4, and Table 5 respectively. These statistical measures were calculated according to the following definitions:

$$\text{Sensitivity} = \frac{TP}{TP+FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2}$$

$$\text{F1} = 2 \text{ x } \frac{Sensitivity \text{ x } Precision}{Sensitivity + Precision} \tag{3}$$

Where TP is is the number of operonic pairs predicted correctly (true positives), FP is the number of non-operonic pairs incorrectly predicted as operonic (false positives), and FN is the number of operonic pairs incorrectly predicted as non-operonic (false negatives).

Table 3. Sensitivity (true positive rate): Percentage of the 730 Operonic gene pairs that were detected by the different tools. With a breakdown of the E. coli gene pairs and the B. subtilis gene pairs.

Tool	All (730 pairs)	E. coli (361 pairs)	B. subtilis (369 pairs)
Operon Hunter	92%	88%	97%
ProOpDB	93%	93%	93%
Door	84%	81%	86%

Table 4. Precision: Percentage of operonic gene pairs predicted by the different tools for both organisms(E. coli and B subtilis), that are true positives.

Operon Hunter	ProOpDB	Door
92%	89%	94%

ProOpDB scores the highest sensitivity but the lowest precision. The opposite is true for Door, that achieves the lowest sensitivity but the highest precision. Operon Hunter's performance is more stable across the two metrics. To unify the cross-comparison over the positive class, we present the corresponding F1 score as a single measure in Table 5. Door has the lowest F1 score, ProobDB comes in second, and Operon Hunter ranks first.

Evaluating Negative-Class Predictions

We present the specificity (true negative rate) to capture the performance over the negative class in table 6. The specificity was calculated using the following definition:

Table 5. F1-score(%) achieved by different tools.

	Operon Hunter	ProOpDB	Door
F1-score	92.1	90.8	88.6

$$\text{Specificity} = \frac{TN}{TN+FP} \tag{4}$$

Where TN is the number of non-operonic pairs predicted correctly (true negatives), and FP is the number of non-operonic pairs incorrectly predicted as operonic (false positives).

Table 6. Specificity (true negative rate): Percentage of the 763 Non-operonic gene pairs that were detected by the different tools. With a breakdown of the E. coli gene pairs and the B. subtilis gene pairs.

Tool	All (763 pairs)	E. coli (461 pairs)	B. subtilis (302 pairs)
Operon Hunter	92%	95%	88%
ProOpDB	89%	90%	88%
Door	95%	94%	97%

Evaluation of Predicted Operon Boundaries

Predicting full operons is a more challenging task that requires the accurate identification of the operon endpoints. Matching earlier reports concerning different methods[7], the accuracies of all 3 tools drop when validated on full operon predictions rather than separate predictions made for every gene pair. We present the percentage of operons predicted fully by each of the tools in Table 7. Full matches are operons reported in the literature that are accurately predicted with the same gene boundaries as the published endpoints. We only consider operons that consist of more than one gene, and were reported in ODB in addition to RegulonDB or DBTBS. This amounted to 254 full operons. Predictions that only partially match a verified operon are not shown. To make full operon predictions, the model starts by generating a prediction for every consecutive gene pair in a genome. In a second pass, pairs predicted as operonic are then merged into full operons. Operon Hunter accurately identifies 85% of the operons fully, which is the most across the 3 tools. Both ProOpDB and Door show a drop in accuracy, to 62% and 61% respectively.

Table 7. Comparison of the results between OperonHunter, ProOpDB, and Door when considering full operon predictions. Exact Operon Matches are the percentage of operons predicted where the endpoints exactly match those of the experimentally verified operons. The percentages are reported over 254 full operons that consist of more than one gene and are reported in RegulonDB/DBTBS and ODB.

	Operon Hunter	ProOpDB	Door
Exact Operon Matches	85%	62%	61%

Discussion

We found that using a neural network that has been pre-trained on a large image dataset, to detect operons using visual representations of genomic regions, achieves better results than the currently available state-of-the-art operon prediction tools. This approach especially demonstrates a clear advantage when it comes to predicting accurate operon boundaries (i.e. predicting operons fully).

We experienced technical difficulties when trying to retrieve the operon predictions made by both tools we are comparing our performance against. Since the tools generate predictions for gene pairs, the shorted possible operon should thus consist of at least two genes. However, this conflicts with the reported predictions made by these tools, as they include operons consisting of only a single gene. We considered single-gene predictions to be part of a non-operonic gene pair. The tools report predictions for only a subset of the genes in a given genome, which leaves many genes without an assigned label. We decided to label the genes with missing predictions as non-operonic, since leaving them out of the performance measures would drastically

diminish the tools' scores. This way, the specificity reported for the tools is the highest they could attain, assuming they would predict all the non-operonic gene pairs correctly.

In a good faith attempt to replicate the work done by ProOpDB, we fed a neural network with the same architecture mentioned in their paper the same features (i.e, the inter-genic distance and STRING score for consecutive pairs of genes). We used our training dataset, which we believe to be richer, considering that they were training on only one organism (E. coli or B. subtilis). We fine tuned different hyper parameters, like the learning rate and the number of epochs, and experimented with different activation functions before settling on the one mentioned in their paper. Instead of generating STRING-like scores for the genes that miss that feature, we followed our previous approach of using a default value of 0.1. It is worth mentioning that the STRING database has been updated since the time of their first publication, and the current version has score assignments for most of the operonic gene pairs (> 99% of the pairs used in our training and testing datasets). Following this approach, we achieved a slightly lower true positive rate, with a slightly higher precision, leading to the same F1 score previously reported for their tool, and a slightly higher specificity, that is still the lowest across the three tools we used in our comparison.

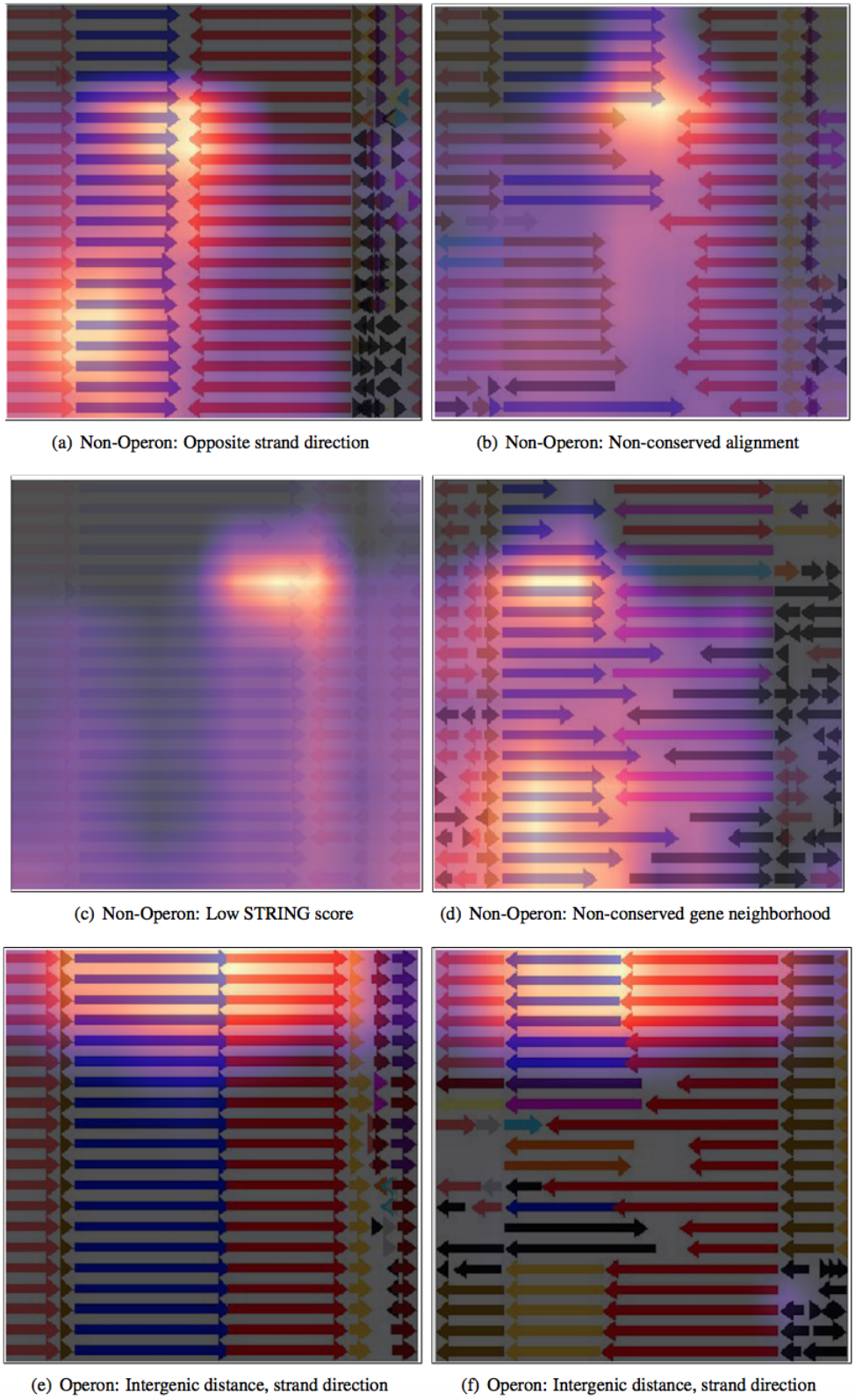
Some methods, including Door, report a decrease in accuracy when the training and testing datasets belong to different organisms. In fact, the accuracy reported by many of the available tools drop anywhere between 11 and 30% when the training data and the operon predictions correspond to different organisms[8,35]. This lack of generalization places severe limitations on the methods' applicability, especially when treating novel genomes. Even for known genomes, this poses the additional challenge of requiring more ground-truth data for operons in other genomes that belong to the same organism. For example, as mentioned earlier, Door switches between a linear model and a more complex decision-tree based model depending on the availability of the experimentally verified operons in the organism of the query genome that can be used for training the model. Our approach alleviates these challenges and generalizes successfully across organisms. As Table 1 shows, the genomes constituting our training dataset span different genera, but the performance of Operon Hunter when predicting operons in both E. coli and B. subtilis does not vary significantly. This is due to the fact that in the compare region viewer service that we use to construct the images, while the genomes aligned with the query genome are chosen based on evolutionary distance, they are eventually represented as images using a more generic method, that captures all underlying relevant operon features.

To help us investigate the model's performance, we used the Grad-Cam[49] method to overlay heat-maps over the input images, highlighting the areas of attention that are affecting the network's decisions the most. Figure 2 shows some of the network's confident predictions. As a first observation, it seems that when predicting a gene pair as non-operonic, the network bases its decision on the entirety of the input image, whereas it mostly focuses on the immediate vicinity of the gene pairs predicted as operonic. Upon closer inspection, we can speculate that:

- in Figure 2(a), the most important feature driving the model's prediction of the input image as a non-operonic gene pair is that pairs' strand directionality, which appears to be in opposition.
- Figure 2(b) seems to highlight the mis-alignment between the genomes, also leading to the prediction of the image as representing a non-operonic gene pair.
- Figure 2(c) has a similar non-operonic class prediction, with the model focusing on the entirety of the image having a low alpha channel, which is a representation of the low STRING score between the query gene pair.
- Figure 2(d) highlights most of the image, with the underlying non-conservation of the query gene pair neighborhood.

All these features were previously mentioned as among the most prominent in operon identification. Operonic gene pairs show more overall conformity across the input image, with flanking regions being mostly aligned/conserved, and query gene pairs having little or no inter-genic distance, and usually similar directions, as shown in Figure 2(e). Even when the neighborhood of the query gene pair is not conserved, as shown in Figure 2(f), the network seems to base its decision on the similar strand direction and lack of an inter-genic distance between the gene pair. Thus, it seems that when predicting gene pairs as operonic, the network focuses on the immediate vicinity of the pair, disregarding other areas of the image. However, the network seems to pick up on anomalies between the gene pair or surrounding them, taking the entirety of the image into consideration when predicting non-operonic images.

In summary, the network seems to be basing its decisions on the features we engineered the input images after, which meets our expectations and supports our general confidence in the method.



(a) Non-Operon: Opposite strand direction

(b) Non-Operon: Non-conserved alignment

(c) Non-Operon: Low STRING score

(d) Non-Operon: Non-conserved gene neighborhood

(e) Operon: Intergenic distance, strand direction

(f) Operon: Intergenic distance, strand direction

Figure 2. HeatMaps generated using the Grad-Cam method and overlayed over input images, highlighting the network's areas of attention that had most influence over the network's decision. Each sub-figure shows the network's correctly predicted label, with what we believe to be the most prominent feature leading to the network's decision.

Author contributions statement

R.A. carried out the implementation and wrote the manuscript. R.S. and F.X. were involved in planning and supervised the work. All authors aided in interpreting the results. All authors provided critical feedback and commented on the manuscript.

Funding

PATRIC has been funded in whole or in part with Federal funds from the National Institute of Allergy and Infectious Diseases, National Institutes of Health, Department of Health and Human Services [HHSN272201400027C]. Funding for open access charge: Federal funds from the National Institute of Allergy and Infectious Diseases, National Institutes of Health, Department of Health and Human Services [HHSN272201400027C].

Availability of data and materials

The datasets generated and/or analyzed during the current study are available in the GitHub repository, https://github.com/ridassaf/OperonHunter

Additional information

The authors declare that they have no competing interests.

References

1. Fran B, Perrin D, Monod J, et al (1960). The operon : a group of genes whose expression is coordinated by an operator. J Bacteriol;29:1727–9.

2. Romero,P.R. and Karp,P.D. (2004) Using functional and organizational information to improve genome-wide computational prediction of transcription units on pathway-genome databases. Bioinformatics, 20, 709–717.

3. Mao, Xizeng, Qin Ma, Chuan Zhou, Xin Chen, Hanyuan Zhang, Jincai Yang, Fenglou Mao, Wei Lai, and Ying Xu. "Door 2.0: presenting operons and their functions through dynamic and integrated views." Nucleic acids research 42, no. D1 (2014): D654-D659.

4. Haller R, Kennedy M, Arnold N, Rutherford R (2010). The transcriptome of Mycobacterium tuberculosis. Appl Microbiol Biotechnol. 2010; 86:1–9.

5. Pelly, S., Winglee, K., Xia, F., Stevens, R. L., Bishai, W. R., & Lamichhane, G. (2016). REMap: operon map of M. tuberculosis based on RNA sequence data. Tuberculosis, 99, 70-80.

6. Price, M. N., Huang, K. H., Alm, E. J., & Arkin, A. P. (2005). A novel method for accurate operon predictions in all sequenced prokaryotes. Nucleic acids research, 33 (3), 880-892.

7. Zaidi, S. S. A., & Zhang, X. (2016). Computational operon prediction in whole-genomes and metagenomes. Briefings in functional genomics, 16 (4), 181-193.

8. Taboada, B., Verde, C., & Merino, E. (2010). High accuracy operon prediction method based on STRING database scores. Nucleic acids research, 38 (12), e130-e130.

9. Taboada, B., Ciria, R., Martinez-Guerrero, C. E., & Merino, E. (2011). ProOpDB: Prokaryotic Operon Data Base. Nucleic acids research, 40 (D1), D627-D631.

10. Taboada, B., Estrada, K., Ciria, R., & Merino, E. (2018). Operon-mapper: a web server for precise operon identification in bacterial and archaeal genomes. Bioinformatics, 34 (23), 4118-4120.

11. Bergman, N. H., Passalacqua, K. D., Hanna, P. C., & Qin, Z. S. (2007). Operon prediction for sequenced bacterial genomes without experimental information. Appl. Environ. Microbiol., 73 (3), 846-854.

12. Fortino, V., Smolander, O. P., Auvinen, P., Tagliaferri, R., & Greco, D. (2014). Transcriptome dynamics-based operon prediction in prokaryotes. BMC bioinformatics, 15 (1), 145.

13. Okuda S, Kawashima S, Kobayashi K, et al (2007). Characterization of relationships between transcriptional units and operon structures in Bacillus subtilis and Escherichia coli. BMC Genomics;8:48.

14. Hodgman TC (2000). A historical perspective on gene/protein functional assignment. Bioinformatics;16:10–5.

15. Joon M, Bhatia S, Pasricha R, et al (2010). Functional analysis of an intergenic non-coding sequence within mce1 operon of M. tu- berculosis. BMC Microbiol;10:128.

16. Wang S, Wang Y, Liang Y, et al (2007). A multi-approaches-guided genetic algorithm with application to operon prediction. Artif Intell Med;41:151–9.

17. Pantosti A, Sanchini A, Monaco M (2007). Mechanisms of antibiotic resistance in Staphylococcus aureus. Future Microbiol;2: 323–34.

18. Yada,T., Nakao,M., Totoki,Y. and Nakai,K. (1999) Modeling and predicting transcriptional units of Escherichia coli genes using hidden Markov models. Bioinformatics, 15: 987–993.

19. Craven,M., Page,D., Shavlik,J., Bockhorst,J. and Glasner,J. (2000) A probabilistic learning approach to whole-genome operon pre- diction. Proc. Conf. Intell. Syst. Mol. Biol., 8: 116–127.

20. Tjaden,B., Haynor,D.R., Stolyar,S., Rosenow,C. and Kolker,E. (2002) Identifying operons and untranslated regions of transcripts using Escherichia coli RNA expression analysis. Bioinformatics, 18 (Suppl. 1): S337–S344.

21. Ermolaeva,M.D., White,O. and Salzberg,S.L. (2001) Predic- tion of operons in microbial genomes. Nucleic Acids Res., 29: 1216–1221.

22. Zheng,Y., Szustakowski,J.D., Fortnow,L., Roberts,R.J. and Kasif,S. (2002) Computational identification of operons in microbial genomes. Genome Res., 12: 1221–1230.

23. Chen,X., Su,Z., Xu,Y. and Jiang,T. (2004) Computational prediction of operons in Synechococcus sp. WH8102. Genome Inform., 15, 211–222.

24. Tran,T.T., Dam,P., Su,Z., Poole,F.L., Adams,M.W., Zhou,G.T. and Xu,Y. (2007) Operon prediction in Pyrococcus furiosus. Nucleic Acids Res., 35, 11–20.

25. Zhang,G.Q., Cao,Z.W., Luo,Q.M., Cai,Y.D. and Li,Y.X. (2006) Operon prediction based on SVM. Comput. Biol. Chem., 30, 233–240.

26. Dam,P., Olman,V., Harris,K., Su,Z. and Xu,Y. (2007) Operon prediction using both genome-specific and general genomic information. Nucleic Acids Res., 35, 288–298.

27. Bockhorst,J., Craven,M., Page,D., Shavlik,J. and Glasner,J. (2003) A Bayesian network approach to operon prediction. Bioinformatics, 19, 1227–1235.

28. Edwards,M.T., Rison,S.C., Stoker,N.G. and Wernisch,L. (2005) A universally applicable method of operon map prediction on minimally annotated genomes using conserved genomic context. Nucleic Acids Res., 33, 3253–3262.

29. Westover,B.P., Buhler,J.D., Sonnenburg,J.L. and Gordon,J.I. (2005) Operon prediction without a training set. Bioinformatics, 21, 880–888.

30. Jacob,E., Sasikumar,R. and Nair,K.N. (2005) A fuzzy guided genetic algorithm for operon prediction. Bioinformatics, 21, 1403–1407.

31. Salgado,H., Moreno-Hagelsieb,G., Smith,T.F. and Collado- Vides,J. (2000) Operons in Escherichia coli: genomic analyses and predictions. Proc. Natl Acad. Sci. USA, 97, 6652–6657.

32. Yan,Y. and Moult,J. (2006) Detection of operons. Proteins, 64, 615–628.

33. Overbeek, R., M. Fonstein, M. D'Souza, G. D. Pusch, and N. Maltsev. (1999). The use of gene clusters to infer functional coupling. Proc. Natl. Acad. Sci. USA 96:2896–2901.

34. Szklarczyk, Damian, John H. Morris, Helen Cook, Michael Kuhn, Stefan Wyder, Milan Simonovic, Alberto Santos et al. "The STRING database in 2017: quality-controlled protein–protein association networks, made broadly accessible." Nucleic acids research (2016): gkw937.

35. Brouwer RWW, Kuipers OP, Van Hijum SAFT (2008). The relative value of operon predictions. Brief Bioinform:367–75.

36. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (2015) ImageNet Large Scale Visual Recognition Challenge. *IJCV*.

37. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, Zbigniew Wojna (2016) Rethinking the Inception Architecture for Computer Vision. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2818-2826.

38. Wattam AR, Davis JJ, Assaf R, Boisvert S, Brettin T, Bun C, Conrad N, Dietrich EM, Disz T, Gabbard JL, Gerdes S, Henry CS, Kenyon RW, Machi D, Mao C, Nordberg EK, Olsen GJ, Murphy-Olson DE, Olson R, Overbeek R, Parrello B, Pusch GD, Shukla M, Vonstein V, Warren A, Xia F, Yoo H, Stevens RL. (2017) Improvements to PATRIC, the all-bacterial Bioinformatics Database and Analysis Resource Center. *Nucleic Acids Res.* **45 (D1)**, D535-D542.

39. Davis JJ, Gerdes S, Olsen GJ, Olson R, Pusch GD, Shukla M, Vonstein V, Wattam AR and Yoo H (2016) PATtyFams: Protein Families for the Microbial Genomes in the PATRIC Database. Front. Microbiol. 7:118. doi: 10.3389/fmicb.2016.00118

40. Rida Assaf, Fangfang Xia, and Rick Stevens (2019). Identifying Genomic Islands with Deep Neural Networks. Bioarxiv.

41. Jia, Y., Weiss, R.J., Biadsy, F., Macherey, W., Johnson, M., Chen, Z. and Wu, Y., 2019. Direct speech-to-speech translation with a sequence-to-sequence model. arXiv preprint arXiv:1904.06037.

42. Poplin, R., Chang, P., Alexander, D., Schwartz, S., Colthurst, T., & Ku, A. et al. (2018) A universal SNP and small-indel variant caller using deep neural networks. *Nature Biotechnology.*

43. Jeremy Howard, (2019) Lesson 2: Deep Learning 2019 - Data cleaning and production; SGD from scratch. Retrieved from https://www.youtube.com/watch?v=ccMHJeQU4Qw

44. How to Retrain an Image Classifier for New Categories | TensorFlow Hub | TensorFlow. (2018). Retrieved from https://www.tensorflow.org/hub/tutorials/image_retraining

45. FastAI | FastAI. (2018). Retrieved from https://docs.fast.ai/index.html

46. Okuda, S. and Yoshizawa, A.C. (2011). ODB: a database for operon organizations, 2011 update. *Nucleic Acids Res.39 (Database issue)*:D552-555.

47. Santos-Zavaleta, Alberto, Heladia Salgado, Socorro Gama-Castro, Mishael Sánchez-Pérez, Laura Gómez-Romero, Daniela Ledezma-Tejeida, Jair Santiago García-Sotelo et al. "RegulonDB v 10.5: tackling challenges to unify classic and high throughput knowledge of gene regulation in E. coli K-12." Nucleic acids research 47, no. D1 (2019): D212-D220.

48. Sierro N., Makita Y., de Hoon M.J.L. and Nakai K. (2008). DBTBS: a database of transcriptional regulation in Bacillus subtilis containing upstream intergenic conservation information. *Nucleic Acids Res.*, 36 (Database issue):D93-D96; doi:10.1093/nar/gkm910.

49. Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE international conference on computer vision (pp. 618-626).