



# A snapshot neural ensemble method for cancer-type prediction based on copy number variations

Md. Rezaul Karim<sup>1,2</sup> · Ashiqur Rahman<sup>2</sup> · João Bosco Jares<sup>1</sup> · Stefan Decker<sup>1,2</sup> · Oya Beyan<sup>1,2</sup>

Received: 18 January 2019 / Accepted: 22 November 2019 / Published online: 30 November 2019  
© The Author(s) 2019

## Abstract

An accurate diagnosis and prognosis for cancer are specific to patients with particular cancer types and molecular traits, which needs to address carefully. The discovery of important biomarkers is becoming an important step toward understanding the molecular mechanisms of carcinogenesis in which genomics data and clinical outcomes need to be analyzed before making any clinical decision. Copy number variations (CNVs) are found to be associated with the risk of individual cancers and hence can be used to reveal genetic predispositions before cancer develops. In this paper, we collect the CNVs data about 8000 cancer patients covering 14 different cancer types from The Cancer Genome Atlas. Then, two different sparse representations of CNVs based on 578 oncogenes and 20,308 protein-coding genes, including genomic deletions and duplication across the samples, are prepared. Then, we train Conv-LSTM and convolutional autoencoder (CAE) networks using both representations and create snapshot models. While the Conv-LSTM can capture locally and globally important features, CAE can utilize unsupervised pretraining to initialize the weights in the subsequent convolutional layers against the sparsity. Model averaging ensemble (MAE) is then applied to combine the snapshot models in order to make a single prediction. Finally, we identify most significant CNVs biomarkers using guided-gradient class activation map plus (GradCAM++) and rank top genes for different cancer types. Results covering several experiments show fairly high prediction accuracies for the majority of cancer types. In particular, using protein-coding genes, Conv-LSTM and CAE networks can predict cancer types correctly at least 72.96% and 76.77% of the cases, respectively. Contrarily, using oncogenes gives moderately higher accuracies of 74.25% and 78.32%, whereas the snapshot model based on MAE shows overall 2.5% of accuracy improvement.

**Keywords** Cancer prediction · Copy number variation · Conv-LSTM network · Convolutional autoencoder · Interpretability · Snapshot ensemble

## 1 Introduction

Cancer results from highly expressed genes due to mutations or alterations in gene regulations that control cell division and cell growth. In such cases, a set of genes called oncogene contribute to conversion of normal cells into a cancerous cells. The change in the structure of occurring genetic aberrations, such as somatic mutations,

copy numbers (CNs), profiles, and different epigenetic alterations, is unique for each type of cancer [8, 45]. As a result, gene expression (GE) can be disrupted by cell division, environmental effects, or genetically inherited from parents. Changes in GE sometimes change the production of different proteins, affecting normal cell behavior. The damaged cells start reproducing more rapidly than usual and gradually increase in the affected area by forming a tumor. Intermittently, such tumors turn into a type of cancer [42, 54]. This is one of the reasons cancer cases are gradually increasing every year and have become the second leading cause of death worldwide.

Consequently, more than 200 types of cancers have been identified, in which each type can be characterized with different molecular profiles requiring unique therapeutic

---

✉ Md. Rezaul Karim  
rezaul.karim@fit.fraunhofer.de

<sup>1</sup> Fraunhofer Institute for Applied Information Technology FIT, Schloss Birlinghoven, Sankt Augustin, Germany

<sup>2</sup> RWTH Aachen University, Aachen, Germany

strategies [45]. According to a statistic from the National Cancer Institute, there were around 14.1 million cancer cases in 2012 in which as many as 8.8 million people died of five leading cancers of lung, liver, colorectal, stomach, and breast [3]. In 2018, an estimated 17.35 million new cases of cancer have been diagnosed in the USA in which 609,640 people died. The number of new cancer cases per year is expected to rise to 23.6 million by 2030, which is anticipated to increase further to 70% by 2035 [46].

With the high-throughput next-generation sequencing technologies, cancer-specific genetic profiling is now possible. The entire genome sequencing data can be used to identify similar genetic mutations and genetic variations associated with different tumors. In particular, copy number variations (CNVs) signify gene or genomic regions that appear in different numbers of copies in different individuals or even in different cells of the same individual, where CN can vary from one person to another by several thousand. About 5–12% of the human genome, including thousands of genes, may be variable in CN, and this variation can be new deletions or duplications of the genome or inherited from the parents by healthy individuals [39], ranging in size from 100 bp to 3 Mb [52].

Although the significance is not fully understood, it is likely that CNVs are responsible for a considerable proportion of phenotypic variation [39]. Such variations may lead to changes in gene dosage and expression [12]. Approximately 179,450 human CNVs were reported in the Database of Genomic Variants (DGV) [20, 52]. Although there are substantially fewer reported CNVs than SNPs, it is estimated that more than 30% of the human genome is covered by at least one CNV compared to the <1% covered by SNPs. Due to the CN changes in DNA segments, GE is changed by disrupting coding sequences, perturbing long-range gene regulations, or altering gene dosages [50]. CNVs result in variations in GE and abnormalities in the human phenotypes [7]. Thus, CNVs are hypothesized to be of functional significance. These changes in GE are responsible for different phenotypic variations or diseases (e.g., disabilities, diabetes, schizophrenia, cancer, and obesity) or envisaged to be associated with other diseases, e.g., autism spectrum disorder [4, 34, 37].

CNVs are also associated with other complex disease susceptibility, which becomes more obvious while analyzing different tumor types, which became more obvious as CNVs are associated with the risk of individual cancer [27, 33, 48] (e.g., risk of pancreatic cancer [19]). For example, CNVs that play a role in genetic predisposition to pancreatic adenocarcinoma [48] are associated with breast cancer risk and prognosis [27] and are responsible for the spatial pattern change in colorectal cancer [33]. The presence of CNVs in cancer patients' cells is abundantly high, which is very different than healthy cells. Consequently,

many clinically important CNVs are outcomes of duplication or deletion of a genomic region with 1 kb (or shorter) length. Therefore, an accurate identification of cancer based on CNVs can help to reveal the genetic predisposition for cancer before cancer actually occurs, so that vigilant prevention and rigorous monitoring may be practiced by those who are highly predisposed.

Previously, CNVs were used to identify minimum redundancy maximum relevance feature selection (mRMR) and incremental feature selection (IFS) methods [53] such that only one sample could be covered at a time or using microarray-based comparative genomic hybridization (aCGH) methods. Then, the extracted CNVs data were used to train machine learning (ML) models for cancer identification and type prediction. These approaches, however, are not capable of simultaneous analysis of multiple samples and recurrent CNVs [32]. In particular, using deep learning (DL) models based on deep neural network (DNN) architectures, such as autoencoders (AEs), more complex and higher-level features can be embedded from the input data and contextual information can be captured [1, 24]. Eventually, learning nonlinear mappings allows transforming and mapped input data space into a lower-dimensional feature space [21]. To emphasize DL-based cancer diagnoses, first we use the MSeq-CNV [32] tool to extract the recurrent CNVs. Since MSeq-CNV uses mixture density for modeling aberrations in depth of coverage and abnormalities in the mate pair insertion sizes, common CNVs across multiple samples can be detected easily.

Since CNVs data are very high dimensional and dynamic, we hypothesize that a neural ensemble method can be effective at analyzing such high-dimensional data through the hierarchical learning process. We train and evaluate Conv-LSTM and convolutional autoencoder (CAE) networks separately, which follow the model averaging ensemble of the classifiers to combine the predictive power of multiple model snapshots from these networks. Further, we provide interpretations (using guided-gradient class activation map (GradCAM++) [6]) about the prediction made by the neural network in order to make the cancer diagnosis more effective and transparent. The contributions of this paper can be summarized as follows:

- We employed MSeq-CNV tool to extract CNV features based on protein-coding genes and oncogenes in order to prepare two rich-labeled training datasets.
- We trained Conv-LSTM and CAE networks to capture both locally and globally important features, where unsupervised pretraining is performed to handle the data sparsity.
- We aggregated the snapshot models from both architectures using MEA to make single predictions.

- We developed a novel approach to generate heat maps (HM) for all the classes sharing prominent pixels across CNV samples and computed the feature importance in terms of mean absolute impact (MAI).
- We identified important and common biomarkers across cancer types and provided interpretations of the predictions made by the models.
- We provided in details the analyses of the outcomes and comparisons with the state-of-the-art approaches.

The rest of the paper is structured as follows: Sect. 2 discusses related works and their potential limitations. Section 3 chronicles the details of data collection and feature engineering process before the network construction and training. Section 4 demonstrates the experimental results and discusses key findings of the study. Finally, explanations of the importance and relevance of the study are reported by highlighting the limitations and discuss some future works before concluding the paper in Sect. 5.

## 2 Related works

For analyzing genomics data and decision making for cancer treatment, different ML and DL algorithms were trained using mixed data types: genomic data, bioimaging data, and clinical outcomes, as shown in Table 1. These approaches are not only proven to be useful at improving cancer prognosis, diagnosis, and treatments, but also revealed the subtypes information about different cancer types [19]. For example, RNA-seq data are used more widely [9, 31, 36] to identify rare and common transcripts, isoforms, and noncoding RNAs in cancer. Other forms of genomic data like single-nucleotide polymorphism (SNP) (i.e., identifies segmental variations across multiple cancer genomes) and array-based DNA methylation sequencing (i.e., identifies epigenetic changes in the

genome) are useful for early genetic changes in cancer, e.g., early-stage and metabolomic detection of ovarian cancer is now possible [14, 45].

Unlike conventional cancer typing methods that work based on morphological appearance analysis, GE levels of the tumor are used to differentiate tumors that have similar histopathological appearances giving more accurate tumor typing results for certain cancer types, e.g., colorectal cancer diagnosis [41]. Further, different types of somatic mutations data such as point mutation, single-nucleotide variation (SNV), small insertion and deletion (INDEL), copy number aberration (CNA), translocation, and CNVs are used. Literature [51] has observed that somatic mutation data are not only associated with complex diseases but also contribute to the growth of different types of cancers. In particular, literature [5] studied CN changes by comparing healthy and cancer-affected patients, which showed that amplification and deletion of certain genes are more common in certain cancer patients than in healthy people.

CNVs analysis based on different statistical methods is also used to identify significant CN associated with different types of cancers. For example, Fisher's exact (FE) test is applied on patient and control groups to identify CN for hereditary breast and ovarian cancer [28]. Although FE is mainly used for CNV analysis [17], ML-based approaches are trending to improve the accuracy of cancer susceptibility, recurrence, and survival prediction [26].

The main challenges are, however, accurate extraction of CNVs and dealing with dimensionality. ML algorithms such as Bayesian networks, SVM, and decision trees are applied effectively to extract the most significant CNVs features for high-dimensional data. In comparison with ML-based approaches, recent DL techniques have shown more accurate and promising results for cancer identification in some studies. In particular, CNN is widely applied [8] on whole slide images in order to detect cancer regions

**Table 1** Different cancer detection methods, data types, and performance

Reference	Approach	Cancer types	#Sample	Data type	Accuracy (%)
Mostavi et al. [36]	CNN	33 primary cancer types	10,340	RNA-seq	94
Karim et al. [22]	DBN/LSTM	14 primary cancer types	15,699	TCGA CNVs	73
Yuan et al. [51]	CNN	25 primary cancer types	14,703	CNA and 3D chromatin	90
Lyu et al. [31]	CNN and GradCAM	33 primary cancer types	10,267	RNA-seq	95.59
Sanaa et al. [13]	LR	6 primary types	3480	CNVs	85
Cruz et al. [8]	CNN	Breast cancer	605	Slide image	96
Danee et al. [9]	MLP	Breast cancer	1210	RNA-seq	94
Ning et al. [53]	Dagging	6 primary types	3480	CNVs	75
Rajana et al. [43]	Deep NN	Prostate cancer	807	Histology	95
Chen et al. [7]	Shallow NN	Colon cancer	590	Gene expression	84

with a very high degree of precision, which is mainly because CNN can extract deep features from different cohorts simultaneously. Literature [9] used a stacked denoising autoencoder to extract features from the RNA-seq data, which are then feed into SVM and shallow ANN to classify malignant or benign tumor of breasts [7]. DeepCNA is another approach proposed for cancer-type prediction based on CNVs and chromatin 3D structure with CNN [51].

Apart from these works, restricted approaches have been proposed based on CNVs for cancer risk and type predictions [11, 13, 53], and literature [11] used recurrent CNVs from non-tumor blood cell DNAs of non-cancer subjects about hepatocellular carcinoma, gastric cancer, and colorectal cancer patients. They were found to reveal the differences between cancer patients and controls with respect to CN losses and CN gains. Although their study can make predictions on the cancer predisposition of an unseen test group of mixed DNAs with high confidence, it is limited to only Caucasian and Korean cohorts.

Ning et al. [53] used CNVs at a level of 23,082 genes for 2916 instances from cBioPortal for Cancer Genomics to classify six different types of cancers, such as breasts, bladder urothelial, colon, glioblastoma, kidney, and head and neck squamous cell. They construct a dagging-based classifier in which the feature space was reduced to CNVs of 19 genes using mRMR and IFS methods. They managed to achieve an accuracy of 75%. Their study indicates that only a few genes may play important roles in differentiating cancer types. Then, Sanaa et al. [13] used the same dataset and extended it by training seven ML classifiers in which random forest algorithm shows 86% accuracy.

In a previous approach [22], we considered segmentation as an important feature because it represents number of CNVs at a DNA location, and the higher the segmentation mean, the higher the CN would be in that region. Then, we calculated the length of a CN and its value based on the difference between the start and end positions of a CNV to extract CNV features. We represented CN loss with negative segmentation means and amplifications of CN with positive segmentation means. CN with segmentation values between a certain range was considered as noise and discarded from rest of the calculation. However, a manual approach for CNV extraction like this often fails to extract non-trivial, high-quality recurrent CNV features in the case of simultaneous analysis of multiple samples [32]. Consequently, MSeq-CNV is used in our study for more efficient extraction of CNVs.

### 3 Materials and methods

In this section, we discuss the data collection, preprocessing, and feature engineering in detail, followed by the preparation of training, validation, and test sets to train the neural networks.

#### 3.1 Data collection

We used the CNVs data from TCGA in our study in which both CN and gene coordinates of different cancer types are considered. These data are hybridized by genome sequencing Affymetrix SNP 6.0 technology, which allows us to examine the largest number of cases along with the highest probe density [40]. Two different types of samples, such as tumor tissue and healthy tissue, are collected from each cancer patient, which are curated from the blood, bone marrow, buccal cell, EBV-immortalized, and solid tissues. For consistency across samples, all data were downloaded from the same platform but with different project ID and gender. However, blood-derived healthy samples of only 14 cancer types were downloaded having at least 400 samples, as shown in Table 2.

We consider gender an important feature for cancer subtyping, since tumors like BRCA and OV are not common in males, whereas PRAD is only common in males. TCGA do not specify if a sample is curated from healthy tissue or tumor tissue, so we downloaded CNV files in separate groups, where each cancer sample is grouped as primary and blood-derived samples during filtering before grouping into male and female. Gene locations in combination with the CN data from all patients were used, where the gene coordinates were collected from cytobands using the ensemble library of `biomaRt` package<sup>1</sup>. Additionally, oncogenes information was downloaded from the TCGA portal.

#### 3.2 Data preprocessing

Some initial reprocessing was required to remove noise and empty string values from both CNV and gene data. For each cancer type, samples were downloaded into four different groups: healthy samples from male, healthy samples from female, tumor samples from male, and tumor samples from female. Then, we preprocessed CNV data for 15,699 separate samples. Then, all the cancer samples and healthy tissue samples were combined. We found that the distribution of CNs across tumor samples is very different, e.g., BRCA has almost double the samples of most other tumors.

<sup>1</sup> <https://www.ensembl.org/biomart>.

**Table 2** Number of CNV samples across 14 different tumor types used in this study

Cohort	#Tumor sample	#Healthy sample	Carcinoma type
COAD	502	468	Colon adenocarcinoma
GBM	609	527	Glioblastoma multiforme
KIRC	586	530	Kidney renal clear cell carcinoma
LGG	514	487	Brain lower-grade glioma
LUAD	554	591	Lung adenocarcinoma
LUSC	524	535	Lung squamous cell carcinoma
OV	572	546	Ovarian serous cystadenocarcinoma
UCEC	548	545	Uterine corpus endometrial carcinoma
BRCA	1103	1103	Breast invasive carcinoma
HNSC	519	562	Head and neck squamous cell carcinoma
THCA	505	513	Thyroid carcinoma
PRAD	501	536	Prostate adenocarcinoma
STAD	442	464	Stomach adenocarcinoma
BLCA	416	397	Bladder urothelial carcinoma

STAD and BLCA have fewer samples compared to other tumor samples except for those having samples between 500 and 600. On the other hand, more than 400 THCA samples have 50–100 CNVs per sample and around 50 OV tumor samples have less than 200 CNVs per sample. Although most tumor samples have CNs between 50 and 400, there are a few samples having more than 1500, giving on average 226.80 CNs per sample. On average, 106.92 CNs exist per healthy sample, which is less than half of tumor samples. Usually, healthy human cells have very different CNs in terms of location, length, and number and vary in humans. This makes the CNVs of the human genome an utterly complex and dynamic structure.

For a cancer-affected patient, this structure becomes even more complex and dynamic as the tumor grows. We solve this dynamic dimension problem by using gene locations of the human genome since DL models expect fixed dimension inputs only. Using MSeq-CNV, we selected a fixed number of genes and extracted the CN that overlapped with the gene locations, removing them from the protein noncoding gene because arguably more than 80% of human genes do not encode any protein, i.e., CNs from these regions have little-to-no effect on the tumor growth. Then inspired by literature [2], we set four thresholds, which divide the CNV spectrum into five regions:

- $-2$  signifies the deletion of both copies,
- $-1$  signifies the deletion of one copy,
- $0$  has exactly two copies, i.e., no CN gain/loss,
- $1$ : denotes a low-level CN gain,
- $2$  denotes a high-level CN gain.

Then, we prepared two different training (i.e., representations) sets based on oncogenes and protein-coding genes used to train the Conv-LSTM and CAE networks.

### 3.3 Feature extraction based on protein-coding genes

We used CNVs about 20,308 protein-coding genes. Since not all of these genes are responsible for tumor growth, weak features from irrelevant genes make the data unnecessarily complex, which might make the neural networks to fail to converge the training, resulting in poor performance. Although using protein-coding genes makes the dimension fixed, 95% of the features will be empty across the samples, e.g., a sample with 500 CNVs has 19,808 empty features making empty features for 19,808 cases. Using gene locations introduced two additional complexities: curse of dimensionality and sparsity.

To solve these problems, we performed CNV analysis in combination with 568 oncogenes responsible for the majority of cancer types. While oncogenes are found to be related to tumor growth, researchers have not confirmed which other genes are also responsible for its growth. In such a setting, consecutive genes with nonzero values represent CN length, and negative or positive segmentation values represent loss and gain, respectively. We hypothesize that Conv-LSTM and CAE networks will be able to identify those hidden nonlinear features.

### 3.4 Feature extraction based on oncogenes

There are structural variations in DNA where protein-coding genes are not present, and these genes have no effect on tumor growth. One way to remove such unnecessary segments is by keeping only those CNs that fall within gene regions and discarding the rest using oncogenes, because oncogenes not only control the cell division but also are responsible for GE changes due to external or natural causes and could lead to the development of tumor



at some point. Thus, by considering only 568 oncogenes, we prepared another CNV representation instead of taking the whole 20,308 genes.

A manual CNV feature extraction is inefficient since recurrent CNVs from multiple samples are often ignored. Thereupon, we used the MSeq-CNV tool [32], which is an efficient CNV extraction method. In order to model the number of mate pairs with aberration in the insertion size and Poisson distribution for emitting the read counts, MSeq-CNV applies the binomial distribution-based mixture density on each genomic position. This applies a mixture density for modeling aberrations in depth of coverage and abnormalities in the mate pair insertion sizes. After applying MSeq-CNV, genome-wide copy gain and loss regions are saved in a matrix form, where each row represents a different CNV call with its sample number, start position, end position, and CN estimation.

An example of the CNV selection process is shown in Fig. 1, where a reference segment of chromosome 1 on top extends from DNA location 109–1145 spanning over two genes: A and B. Below this reference, there are six candidate CNs (C1–C6) at this location. C1 is a candidate CN that extends between genes A and B, but C1 does not overlap with any one of them, which is why we remove it from the calculation. Candidates C2 and C3 are CNs, too, which slightly overlap with A and B, respectively. We still consider them as a valid CN because gene positions are not always fixed. C4 and C5 both fully overlap with genes A and B; hence, they are also valid CNs. C6 is not only a big segmental CN, but it also overlaps with both genes. However, since we consider only the part of genes that overlap, we discard rest of the regions from the calculation. The calculation is based on the hypothesis that if a candidate CN overlaps with a gene even just by one base pair, we still consider it as a valid CN [35].

Each gene has a start and end positions in DNA, which is similar to CNs having starting and ending positions. For each gene, MSeq-CNV checks whether a sample has any CNV region overlapping with that gene. Then, it assigns the segment value for that gene; otherwise, 0 is assigned, i.e., a tumor sample has different CN variations than a reference, which is measured by the segment value. If the value is positive, CN in that gene is considered to be more increased than the reference. A negative value, however, signifies that the gene was deleted from that DNA location. A sample of the final training data is shown in Table 3. The first row shows a male patient’s GBM tumor sample, where genes PRDM16 and RPL22 have CN losses with segmentation values of 0.3195 and 0.2154, respectively, whereas gene CAMTA1 has no CNVs. Eventually, our training data dimension reduced from 20,308 to 568 and the label column.

### 3.5 Network constructions and training

We construct and train the Conv-LSTM and CAE networks using two different CNVs representations based on oncogenes and protein-coding genes, in which multiple snapshot models are created. We evaluate each trained model independently, which is followed by the model averaging ensemble (MAE) to combine the snapshot models of both networks.

However, to apply the convolutional (conv) operations and to find important biomarkers using a guided-gradient class activation map (aka. GradCAM), we embedded each CNVs sample into a 2D image. Each protein-coding gene-based CNV sample was reshaped from a  $20,308 \times 1$  array into a  $144 \times 144$  image, whereas each oncogene-based CNV sample was reshaped from a  $568 \times 1$  array into  $24 \times 24$  image by adding zero padding around the edges. Then, all the images are normalized to [0,255] without losing any information.

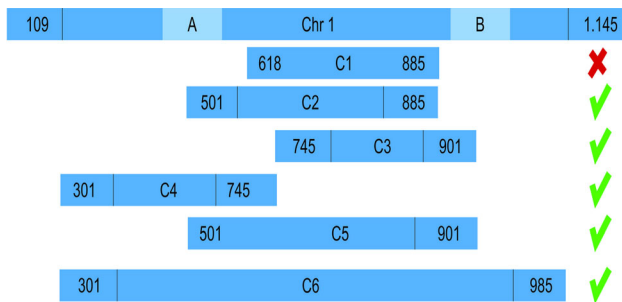
#### 3.5.1 Conv-LSTM network

We construct and train the Conv-LSTM network by combining CNN and LSTM layers in a single architectures as shown in Fig. 2. While the CNN uses conv filters to capture local relationship values, an LSTM network can carry overall relationship of a whole CNV sequence more efficiently. This turns Conv-LSTM into a powerful architecture to capture long-term dependencies between features extracted by CNN. To do so, all the input  $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_t$  cells in Conv-LSTM output  $\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_t$ , hidden states  $\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_t$ , and gates  $i_t, f_t, o_t$  of the network as 3D tensors whose last two dimensions are spatial dimensions.

Then, the Conv-LSTM determines the future state of a certain cell in the input hyperspace by the inputs and past states of its local neighbors, which is achieved by using a convolution operator in the state-to-state and input-to-state transition as follows [49]:

$$\begin{cases} i_t = \sigma(W_{xi} * \mathcal{X}_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \circ \mathcal{C}_{t-1} + b_i) \\ f_t = \sigma(W_{xf} * \mathcal{X}_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \circ \mathcal{C}_{t-1} + b_f) \\ \mathcal{C}_t = f_t \circ \mathcal{C}_{t-1} + i_t \circ \tanh(W_{xc} * \mathcal{X}_t + W_{hc} * \mathcal{H}_{t-1} + b_c) \\ o_t = \sigma(W_{xo} * \mathcal{X}_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \circ \mathcal{C}_t + b_o) \\ \mathcal{H}_t = o_t \circ \tanh(\mathcal{C}_t) \end{cases} \tag{1}$$

In Eq. 1, ‘\*’ denotes the conv operation and ‘o’ is the entrywise multiplication of two matrices of same dimensions. Intuitively, an LSTM layer treats an input feature space as timesteps and outputs arbitrary hidden units per timestep. The second LSTM layer emits an output ‘H,’ which is then reshaped into a feature sequence to feed into fully connected layers to predict the cancer types at the



**Fig. 1** An example CNV selection process from a set of candidate CNs from a reference segment of chromosome 1 that extends from one DNA location to another location

next step, which is again used as an input at the next time step. Once the input feature space is passed to the conv layer, the input is padded such that the output has the same length as the input.

Output of each conv layer is then passed to the dropout layer to regularize learning to avoid overfitting [25]. This involves the input feature space into lower-dimensional representation, which is then further down-sampled by two different max pooling layers and a max pooling layer (MPL) by setting the pool size, where the output of an MPL can be considered as an ‘extracted feature’ from each 2D CNV image. Since each MPL follows to ‘flatten’ the output space by taking the highest value in each

timestep dimension, this helps produce a sequence vector (i.e., feature sequence) from the last LSTM layer, which will hopefully force the CNVs of specific genes that are highly indicative of being responsible for specific cancer type. Then, this vector is fed into the neural network after passing through another dropout layer and a fully connected Softmax for the probability distribution over the classes.

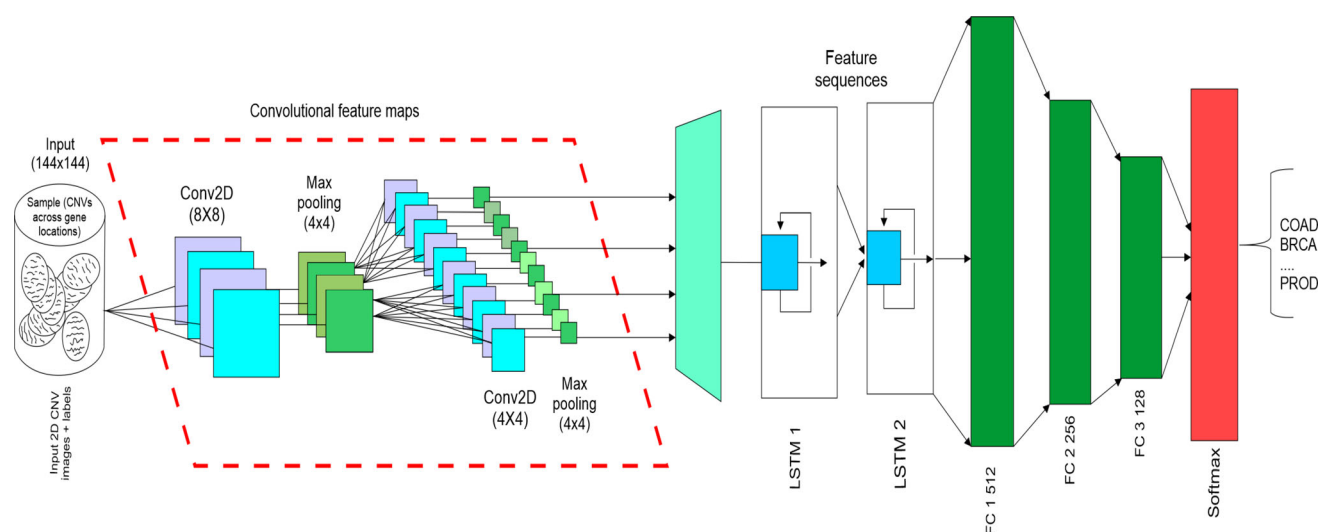
### 3.5.2 Convolutional autoencoder classifier

Using the Conv-LSTM-based network, we have seen how to extract both local and globally important features for classifying individual samples from a limited number of labeled samples. However, in cases of very small numbers of training samples, unsupervised pretraining has proven highly effective [10, 16, 21, 47]. Further, since both datasets are very sparse, we hypothesize that using a CAE-based representation learning (with reduced dimension) and classification scheme should further enhance the classification accuracy.

The architecture of the CAE used in this work consists of a 19-layer CAE. A batch normalization layer is placed after every conv layer to normalize the inputs to the next layer in order to fight the internal covariate shift problem. The ReLU activation function is used in every layer except

**Table 3** Example of training data prepared based on CNVs representation of oncogenes

Type	Gender	PRDM16	RPL22	CAMTA1	MTOR	...	MTCP1
1	1	− 0.3195	− 0.2154	0	0.4767	...	0.652
2	0	0.230	− 0.552	1.715	− 0.92	...	− 1.0
3	0	− 1.240	0.975	0.350	0.642	...	0.985



**Fig. 2** The schematic representation of the Conv-LSTM network, which starts from taking input CNV samples and passing to CNN layers before getting a sequence vector representation from the LSTM layers to pass to dense, dropout, and softmax layers for the cancer-type prediction

for the last layer that uses a softmax activation function. From the given CNV samples, a conv layer in the encoder calculates the feature map. The pooling layer is calculated by down-sampling the conv layer by taking the maximum value in each non-overlapping subregion. The CAE classifier contains two parts: (i) the CAE and (ii) the classifier. The CAE part has the following structure (Fig. 3):

- Input: each CNVs sample is reshaped to  $144 \times 144$
- Convolutional layer: of size  $32 \times 144 \times 144$
- Batch normalization layer: of size  $32 \times 20, 736$
- Convolutional layer: of size  $32 \times 20, 736$
- Batch normalization layer: of size  $32 \times 10, 368$
- Max pooling layer: of size  $32 \times 10, 368$
- Convolutional layer: of size  $64 \times 10, 368$
- Batch normalization layer: of size  $64 \times 10, 368$
- Convolutional layer: of size  $64 \times 10, 368$
- Batch normalization layer: of size  $64 \times 10, 368$
- Convolutional layer: of size  $128 \times 10, 368$
- Batch normalization layer: of size  $128 \times 10, 368$
- Convolutional layer: of size  $128 \times 10, 368$
- Batch normalization layer: of size  $128 \times 10, 368$
- Convolutional layer: of size  $128 \times 10, 368$
- Batch normalization layer: of size  $128 \times 10, 368$
- Convolutional layer: of size  $64 \times 10, 368$
- Batch normalization layer: of size  $64 \times 10, 368$
- Upsampling layer: of size  $64 \times 20, 736$
- Convolutional layer: of size  $1 \times 20, 736$ .

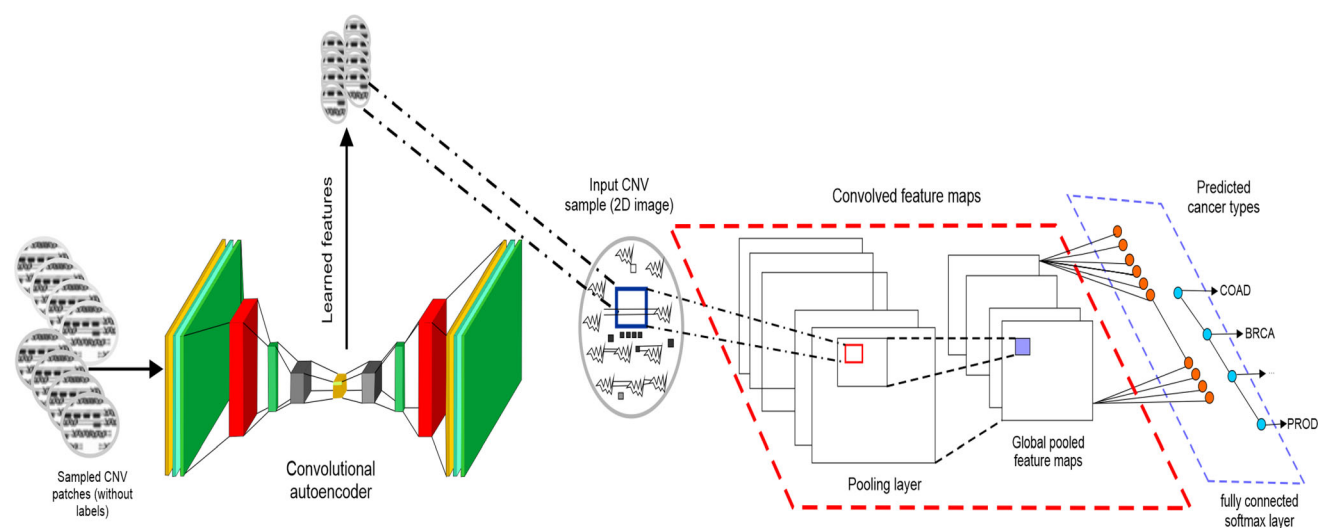
After training the CAE, we remove the decoder components by making the first 19 layers trainable false, since the encoder part is already trained. On top of these components, we add a flattening layer, followed by a fully connected dense layer of size 128, followed by the softmax output unit of 14, i.e., number of classes.

### 3.5.3 Ensemble of classifiers

The model ensemble technique helps the neural network to achieve improved performance compared to the predictions from a single model by reducing the generalization error. It can be achieved by training multiple model snapshots during a single training run and combining their predictions to make an ensemble prediction called snapshot ensemble [18]. A limitation of this approach, however, might be that the saved models will be similar, resulting in similar predictions and prediction errors. This will not offer much benefit from combining their predictions unless we have already introduced diversities during model training [18].

This issue can be addressed by changing the learning algorithm to force the exploration of different network weights during a single training run that will result, in turn, with models that have differing performance. One way to achieve this is using cyclic annealing cosine (CAC), which aggressively but systematically changes the learning rate (LR) over training epochs to produce very different network weights [30]. The CAC approach requires total training epochs, maximum learning rate, and number of cycles, as well as the current epoch number making the initial LR and the total number of training epochs as two hyperparameters. Therefore, CAC will have an effect of starting with a large LR, which is relatively rapidly decreased to a minimum value before being dramatically increased again to the following LR for the given epoch [18]:

$$\alpha(t) = \frac{\alpha_0}{2} \left( \cos \left( \frac{\pi \bmod (t-1, \lceil T/M \rceil)}{\lceil T/M \rceil} \right) + 1 \right) \quad (2)$$



**Fig. 3** The schematic representation of the CAE classifier, which starts from taking input CNV samples and passing to CNN layers before getting a sequence vector representation to pass to dense, dropout, and softmax layers



In Eq. 2,  $\alpha(t)$  is the LR at epoch  $t$ ,  $\alpha_0$  is the maximum LR,  $T$  is the total epoch,  $M$  is the number of cycles,  $mod$  is the modulo operation, and square brackets indicate floor operations. After training the network for  $M$  cycles, best weights at the bottom of each cycle are saved as the snapshot of each model, which gives us  $M$  model snapshots. The model converges to a minimum at the end of training. The model undergoes several LR annealing cycles, converging to and escaping from multiple local minima during snapshot ensembling.

The overall process is outlined in Fig. 4. Finally, all the snapshots are collected together and used in the final ensemble using MEA technique as shown in Fig. 5. Although each model’s weights are subjected to the dramatic changes during training for the subsequent LR cycle, CAC allows the learning algorithm to converge to a different solution.

### 3.5.4 Networks training

We train the networks in two steps: (i) First, we train each network independently and (ii) we create multiple snapshots of both networks before applying the MAE. While training the Conv-LSTM and CAE networks, network parameters were initialized with Xavier initialization [15] and trained using a first-order gradient-based optimization technique called AdaMax to optimize the categorical cross-entropy loss shown in Eq. 3 of the predicted cancer type  $P$  vs. actual cancer type  $T$ :

$$E_c = \sum_{i,j,k} T_{i,j,k} \log P_{i,j,k} + (1 - T_{i,j,k}) \log(1 - P_{i,j,k}) \quad (3)$$

On the other hand, the Softmax activation function is used in the output layer for the probability distribution over the

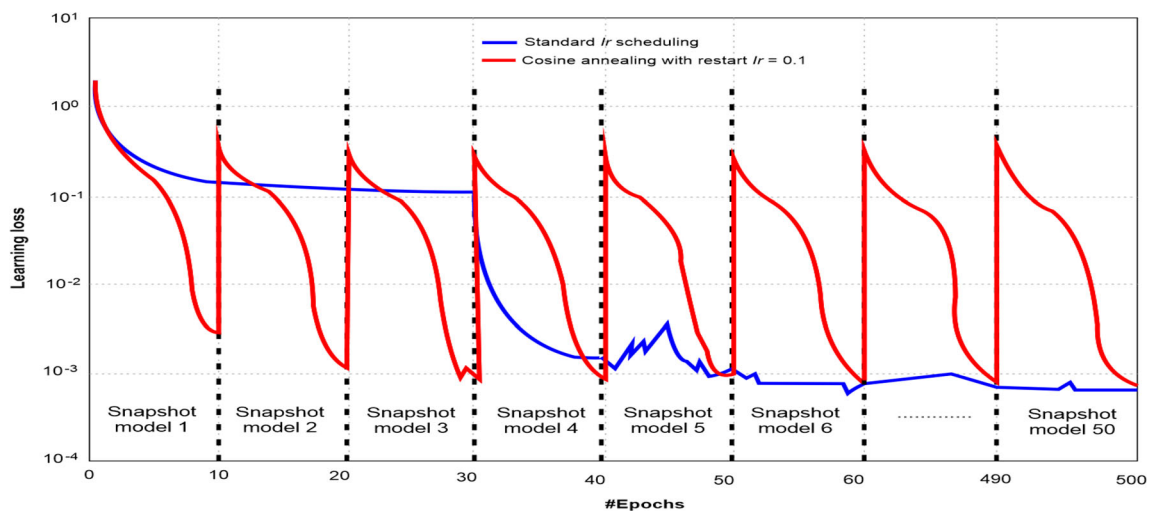
classes. The hyperparameters are defined in grid search and tenfold cross-validation by varying learning rates and different batch sizes. Further, we observe the performance by adding the Gaussian noise layers followed by Conv and LSTM layers to improve model generalization and reduce overfitting. When we train the ensemble model, we set total epochs, maximum learning rate, number of cycles, and the current epoch number, but we make the initial learning rate and the total number of epochs two hyperparameters as expected by the CA approach.

$$h_{MAE} = \frac{1}{m} \sum_0^{m-1} h_{M-i}(\mathbf{x}) \quad (4)$$

After training each network for certain cycles, the best weights are saved, giving model snapshots. The ensemble prediction at test time is the average of the last  $m$  model’s outputs, where  $m \leq M$ . If  $\mathbf{x}$  is a test sample and  $h_i(\mathbf{x})$  is the softmax score of snapshot  $i$ , the final output is the simple mean of the last  $m$  models shown in Eq. 4, giving the lowest test error [18].

### 3.5.5 Finding and validating important biomarkers

Since the data are very high dimensional, we chose not to go for manual feature selection. Rather, we let both CAE and Conv-LSTM networks extract the most important features inspired by literature [23]. The guided backpropagation helps to generate more human-interpretable but fewer class-sensitive visualizations than the saliency maps (SM) [38]. Since SM use true gradients, the trained weights are likely to impose a stronger bias toward specific subsets of the input pixels. Accordingly, class-relevant pixels are highlighted rather than producing random noise



**Fig. 4** Training loss of Conv-LSTM network using standard learning rate (blue) and  $M = 50$  cosine annealing cycles (red). The intermediate models, denoted by the dotted lines, form an ensemble at the end of training

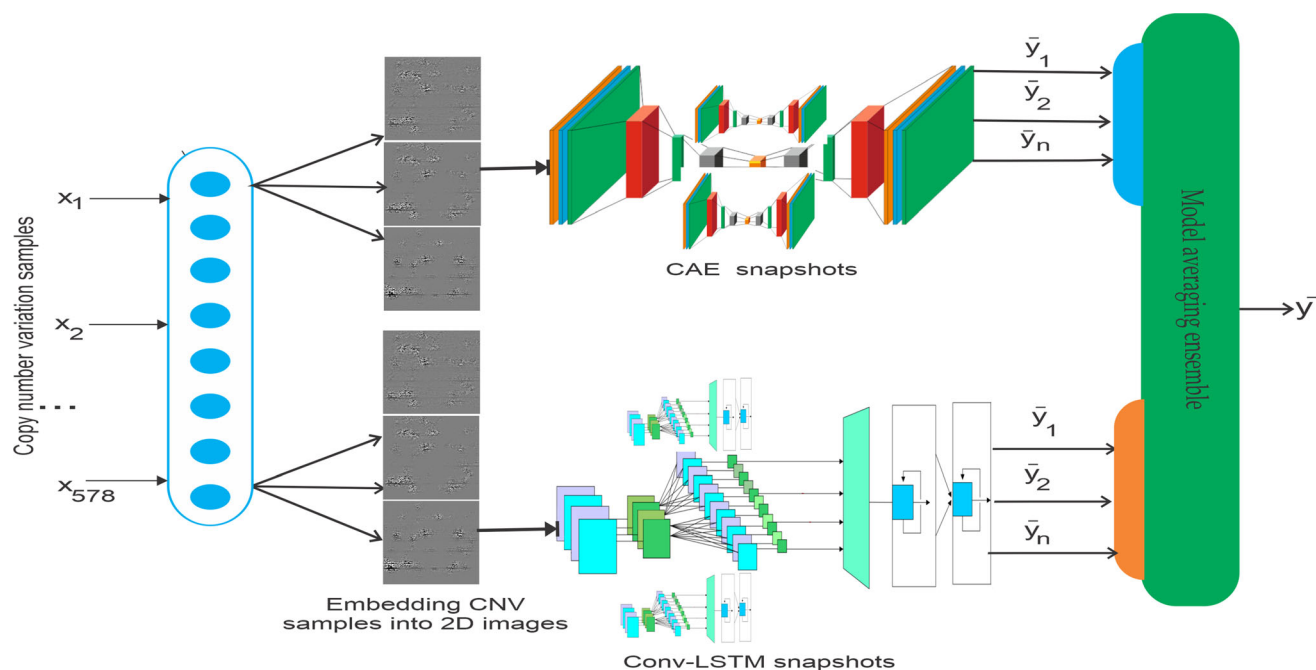


Fig. 5 Model averaging ensemble of Conv-LSTM and CAE snapshots

[38]. Therefore, GradCAM++ is used to draw the HM to provide attention to most important genes. Class-specific weights of each FM are collected from the final conv layer through globally averaged gradients (GAG) of FMs instead of pooling [6]:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{5}$$

where  $Z$  is the number of pixels in a FM,  $c$  is the gradient of the class, and  $A_{ij}^k$  is the value of  $k$ th FM at  $(i, j)$ . Having gathered relative weights, the coarse SM,  $L^c$  is computed as the weighted sum of  $\alpha_k^c * A_{ij}^k$  of the ReLU activation function and employ the linear combination to the FM, since only the features with positive influence on the class are of interest [6].

$$L^c = ReLU \left( \sum_i \alpha_k^c A^k \right) \tag{6}$$

The GradCAM++ replaces the GAG with a weighted average of the pixel-wise gradients (Eq. 7), since the weights of pixels contribute to the final prediction (Eq. 8) by aggregating Eq. 7 and  $\alpha_{ij}^{kc}$  (Eq. 9). In summary, it applies the following iterators over the same activation map  $A^k$ ,  $(i, j)$  and  $(a, b)$ :

$$w_k^c = \sum_i \sum_j \alpha_{ij}^{kc} \cdot ReLU \left( \frac{\partial y^c}{\partial A_{ij}^k} \right) \tag{7}$$

$$y^c = \sum_k w_k^c \cdot \sum_i \sum_j A_{ij}^k \tag{8}$$

$$\alpha_{ij}^{kc} = \frac{\frac{\partial^2 y^c}{(\partial A_{ij}^k)^2}}{2 \frac{\partial^2 y^c}{(\partial A_{ij}^k)^2} + \sum_a \sum_b A_{ab}^k \frac{\partial^3 y^c}{(\partial A_{ij}^k)^3}} \tag{9}$$

**Algorithm 1:** Computing feature importance and ranking genes

**Input :** 2D CNV images  $\mathcal{D} = (d_1, d_2, \dots, d_n)$  having ground truth (i.e., labels)  $\mathcal{L} = (l_1, l_2, \dots, l_j)$  on which a CNN model is trained for each fold  $\mathcal{M} = (m_1, m_2, \dots, m_i)$  to find  $k$  for top- $k$  genes that satisfy MAI threshold.  
**Output:** feature importance  $\mathcal{F} = (f_1, i_1)(f_2, i_2), \dots, (f_n, i_n)$  and top features  $\mathcal{T}$  across all images per fold per class.

```

for fold ∈ FOLDS do
    P ← {} // Guided backprop for each image per fold per class
    K ← {} // GradCAM++(GCAM) for all images per fold per class
    I ← {} // GCAM of each image in a fold
    G ← {} // GCAM for all images per fold per class
    F ← {} // Feature importance of each gene per class per fold
    T ← {} // Top genes and importance per class per fold
    for d ∈ D do
        K ← gradCAM++(m_d, d, l_d) // GCAM of images per fold per class
        P ← guidedBackprop(m_d, d) // Guided backprop of each image
        I ← K * P // GCAM of each image
        G ← G ∪ I // GCAM for all the images in fold
    F ← 1/N ∑_{i=1}^N G // Mean absolute impact for genes for axis=0
    if F_i < σ then
        F ← F - F_i // If the feature importance is less than MAI
        T ← sort_k(F) // Pop off insignificant genes
    // Sort and choose top genes based on MAI
Return F, T

```

**Algorithm 2:** Identification of important areas

**Input :** importance of current class across folds  $\mathcal{F} = (f_1, \dots, f_i)$ , height  $h$  & width  $w$  of rectangle, and MAI threshold  $\sigma$ .  
**Output:** important areas  $\mathcal{C} = (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$  in an image per fold.

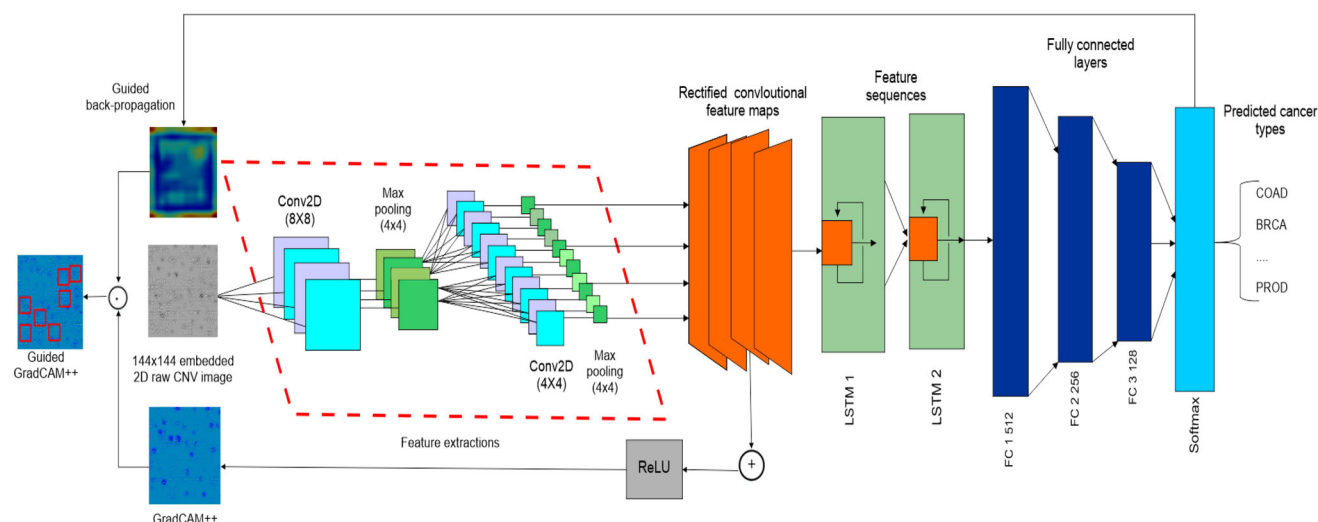
```

for fold ∈ FOLDS do
    A ← dict() // Importance of areas
    S ← list() // Sorted areas by MAI
    C ← list() // Important areas
    for h do
        for w do
            area ← F[h : h + shape[0], w : w + shape[1]] // Area of image
            impA ← 0 // Importance of current area in the image
            for row ∈ area do
                for imp ∈ row do
                    if imp > σ then
                        impA += imp - σ // If feature importance is greater than MAI
                        // Importance of area = current importance - MAI
            A[area] = impA // We update the importance of the area
    S ← sort(A, reverse = true)
    for a, i ∈ S do
        if a ∩ i = then
            C ← C ∪ a // Non-intersecting area with important areas
            // It's a new important area added to the list
Return C

```

Consequently, they can highlight class-relevant pixels rather than produce random noise [38]. This further motivated us to use the GradCAM++ [6] to draw the heat map to provide attention to the most important genes and related CNV in their locations. Since GradCAM++ requires all the samples to run through the neural network once, we let the trained CAE model set and record the activation maps in the forward pass, and the gradient maps in the back-propagation collect the heat map for each sample from the trained model.

The idea is also to find most important biomarkers by ranking them based on mean absolute impact (MAI) threshold. For a batch of samples, first we compute the loss for a column across the samples. Then, we shuffled that column and compute the loss again. The feature importance is then given by the difference between the loss obtained when the respective column is shuffled and the original loss. Then, the MAI is computed as the mean of the  $F_i$  across the batches. The idea is that when the column is shuffled, the relationship between that feature and the output of the model is broken. Therefore, if the loss



**Fig. 6** Schematic representation of important biomarkers identification, which starts from taking a raw CNV sample and passing to conv layers before obtaining rectified conv feature maps to be used to set

increases a lot, making the model predictions are less accurate, then there is high chance that that variable could be an important predictor. Conversely, if the loss remains almost equal, that predictor is not contributing significantly toward increasing or decreasing the accuracy. Hence, it can be considered as a less important feature.

The second approach we could employ is by first computing the gradient for a class for the last conv layer and using the ReLU activation function to normalize the gradient between 0 and 1, which is the importance. However, if the gradient is negative, we would have to change it to 0, which would force to keep track of weak CNV features. Algorithms 1 and 2 show the pseudocode used for computing feature importance with ranking genes and for identification of the important areas on the heat maps, while Fig. 6 shows the schematic representation of both step.

### 3.5.6 Hyperparameter tuning

Since an appropriate selection of hyperparameters can have a huge impact on the performance of deep architecture, we perform the hyperparameter optimization through a random search and fivefold cross-validation tests. First, we optimize the parameters for the individual model, followed by the optimization of the ensemble model. While training the Conv-LSTM and CAE, in each of five runs, 70% of the data are used for the training, 30% of the data are used for the evaluation, and 10% of data from the training set are used for the validation of networks to optimize the learning rate, batch size, number of filters, kernel size, dropout probability, and Gaussian noise probability.

However, for pretraining the CAE, 90% of data are used for the training and 10% of the data are used for validation.

and record the activation maps in the forward pass, and gradient maps in the backpropagation to put attention using heatmap

When we train the ensemble model, we set number of epochs (NE) to 500, maximum learning rate to 1.0, and number of cycles (NC) to NE/10, giving 50 snapshots for each model for 50 cycles through a grid search, since an exhaustive optimization would be too computationally expensive. Further, we only use a single object property to test how results change with each choice of parameter, due to computational constraints.

However, to get the class-specific heat map, we average all the normalized heat maps from the same class, as suggested in the literature [44]. In the heat map, a higher intensity pixel represents a higher significance to the final prediction, which indicates higher importance of its corresponding gene and its CNV. Finally, top genes are selected based on the intensity rankings in the heat maps as described in Algorithms 1, 2, and 6.

## 4 Experiment results

We carried out several experiments based on protein-coding genes and oncogenes. By using each network and CNV representation, cancer-type predictions were experimented separately. Results of each experiment with a comparative analysis will be discussed in this section.

### 4.1 Experiment setup

All programs were implementation in Python<sup>2</sup>. The software stack comprised Scikit-learn and Keras with the TensorFlow backend. The network was trained on a

<sup>2</sup> <https://github.com/rezacsedu/Neural-ensemble-cancer>.

**Table 4** Performance on cancer-type prediction using noncoding genes

Tumor type	Conv-LSTM (72.96%)		CAE (76.77%)		Support
	Precision	Recall	Precision	Recall	
COAD	0.6981	0.7325	0.7924	0.8250	133
GBM	0.8127	0.8275	0.8247	0.8765	151
KIRC	0.8013	0.7335	0.8331	0.8468	150
LGG	0.8250	0.6935	0.8642	0.7835	120
LUAD	0.6835	0.6584	0.6992	0.7238	136
LUSC	0.6937	0.7294	0.7438	0.8352	132
OV	0.7539	0.8356	0.8365	0.7613	145
UCEC	0.6645	0.6829	0.7435	0.6834	131
BRCA	0.7357	0.7587	0.7954	0.8386	269
HNSC	0.7134	0.7235	0.6853	0.7542	141
THCA	0.6967	0.6959	0.7025	0.7654	121
PRAD	0.7225	0.7934	0.8395	0.8345	124
STAD	0.6528	0.6917	0.6776	0.6512	113
BLCA	0.7356	0.6589	0.7925	0.7489	108
Avg/total	0.7278	0.7296	0.7593	0.7677	1974

computer with an i7 CPU core, 32GB of RAM, Ubuntu 16.04 OS, and Nvidia GTX 1050i GPU with CUDA and cuDNN enabled to make the overall pipeline faster. Results based on the best hyperparameters produced through random search are reported empirically.

We report the results using macro-averaged precision and recall since classes are imbalanced. We did not report F1-scores since it is significant only when the value of precision and recall is very different. Further, for cancer

diagnosis, it is important to have both high precision and recall. Hence, results with very different precision and recall are not useful in cancer diagnosing and tumor-type identification. Finally, the MAE is used to report the final predictions.

#### 4.2 Performance analysis of individual model

Classification accuracies using oncogenes and all noncoding genes vary. In particular, using protein-coding genes, classifiers perform moderately well, giving accuracies of 72.96% and 76.77% with Conv-LSTM and CAE network, respectively. Since the classes are imbalanced, only the accuracy will give a very distorted estimation of the cancer types. Class-specific classification reports are thus shown in Tables 4 and 5.

As shown in Table 4, precision and recall for the majority of cancer types were moderately high in which CAE performed mostly better than Conv-LSTM network. In particular, CAE classifier can classify COAD, GBM, KIRC, BRCA, LUSC, and PRAD more confidently (at least 82.50% of the cases). On the contrary, Conv-LSTM classified the OV tumor samples more accurately than CAE classifier (83.56% vs 76.13%). The downside is that both classifiers have made a substantial amount of mistakes, e.g., CAE can classify STAD and UCEC tumor cases only 65% and 68% accurately, whereas the Conv-LSTM network made more mistakes particularly for the STAD, BLCA, THCA, UCEC, LUAD, and LGG tumor samples.

Although the tumor-specific classification accuracies varied across classes, the overall accuracies increased moderately and reached 74.25% and 78.32% using Conv-

**Table 5** Performance on cancer-type prediction using oncogenes

Tumor type	Conv-LSTM (74.67%)		CAE (78.32%)		Support (CAE)
	Precision	Recall	Precision	Recall	
COAD	0.7785	0.7564	0.8265	0.8193	133
GBM	0.8254	0.8330	0.8433	0.8524	151
KIRC	0.7753	0.7835	0.8172	0.8035	150
LGG	0.8235	0.7136	0.7976	0.8031	120
LUAD	0.7520	0.7351	0.7584	0.7674	136
LUSC	0.7528	0.6475	0.7139	0.7256	132
OV	0.8327	0.8407	0.8665	0.8538	145
UCEC	0.7726	0.7625	0.8732	0.8412	131
BRCA	0.7956	0.8075	0.8012	0.7965	269
HNSC	0.8253	0.6932	0.7025	0.6929	141
THCA	0.7953	0.8025	0.7429	0.7581	121
PRAD	0.8267	0.8237	0.8781	0.8626	124
STAD	0.7924	0.6945	0.7632	0.7737	113
BLCA	0.8368	0.7979	0.8623	0.8553	108
Avg/total	0.7543	0.7467	0.7945	0.7832	1974



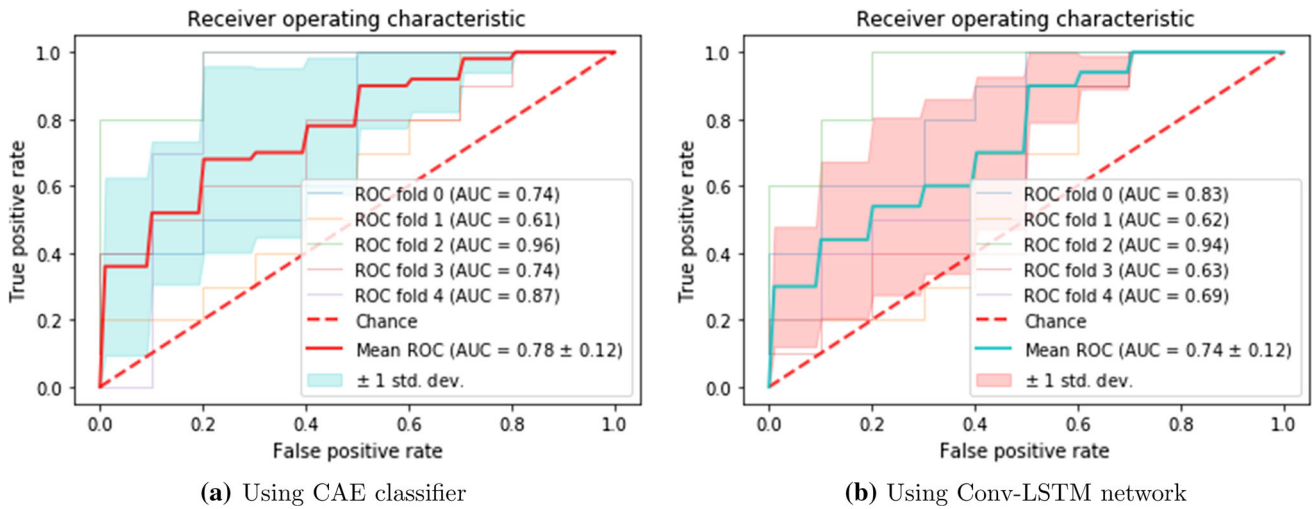


Fig. 7 The ROC curves for Conv-LSTM and CAE models across folds

LSTM and CAE, respectively. As shown in Table 5, precision and recall scores for most of the cancer types are higher than that of protein-coding gene-based CNVs. Similar to the previous experiment, CAE performed mostly better than the Conv-LSTM network across tumor types. In particular, the CAE classifier can correctly classify COAD, GBM, OV, UCEC, PRAD, and BLCA in at least 80% of the cases showing higher confidence.

On the contrary, Conv-LSTM classified the BRCA and THCA tumor samples more accurately than the CAE classifier. The downside is that both classifiers have made substantial mistakes, too. For example, CAE could classify HNSC and LUSC tumor samples accurately in only 69%

and 72% of the cases, respectively, whereas the Conv-LSTM network made more mistakes, particularly for the STAD, HNSC, LUSC, and LGG tumor samples. In summary, both classifiers performed moderately well except for certain types of tumor cases such as STAD, HNSC, BLCA, THCA, UCEC, LUAD, LUSC, and LGG. The ROC curves in Fig. 7 based on CNVs from oncogene show that AUC scores generated by both the Conv-LSTM model and CAE classifier are consistent across folds, and AUC scores generated by the CAE classifier are about 4% better than that of the Conv-LSTM network.

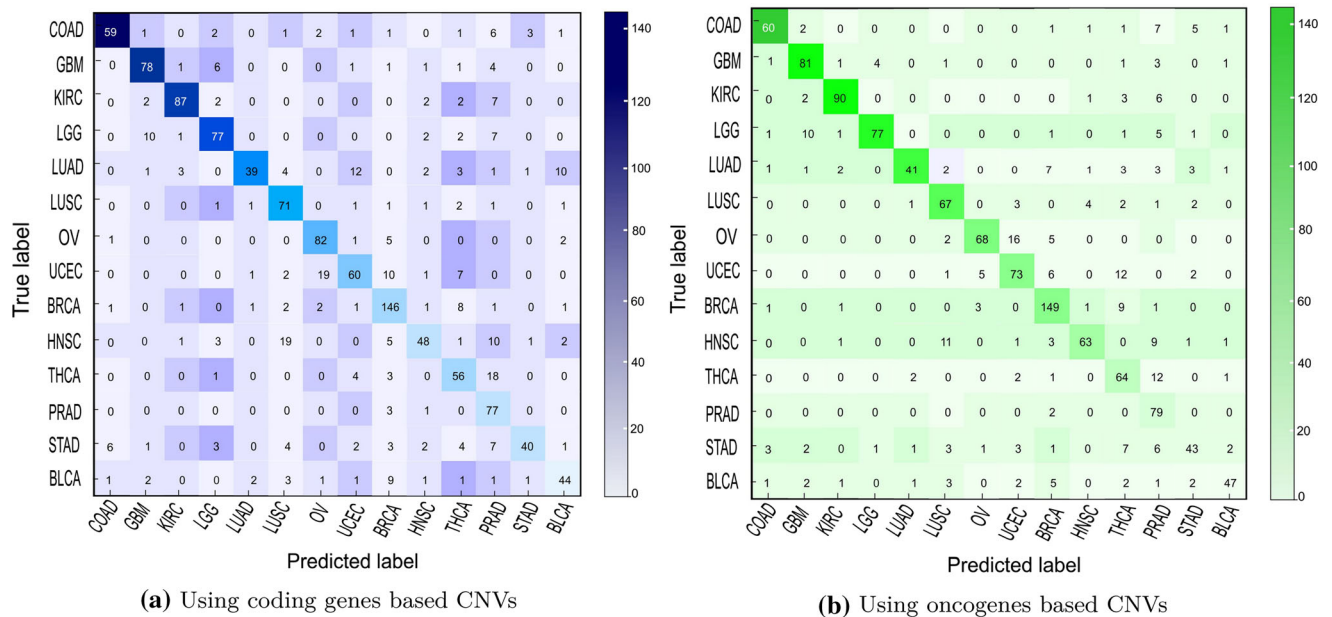
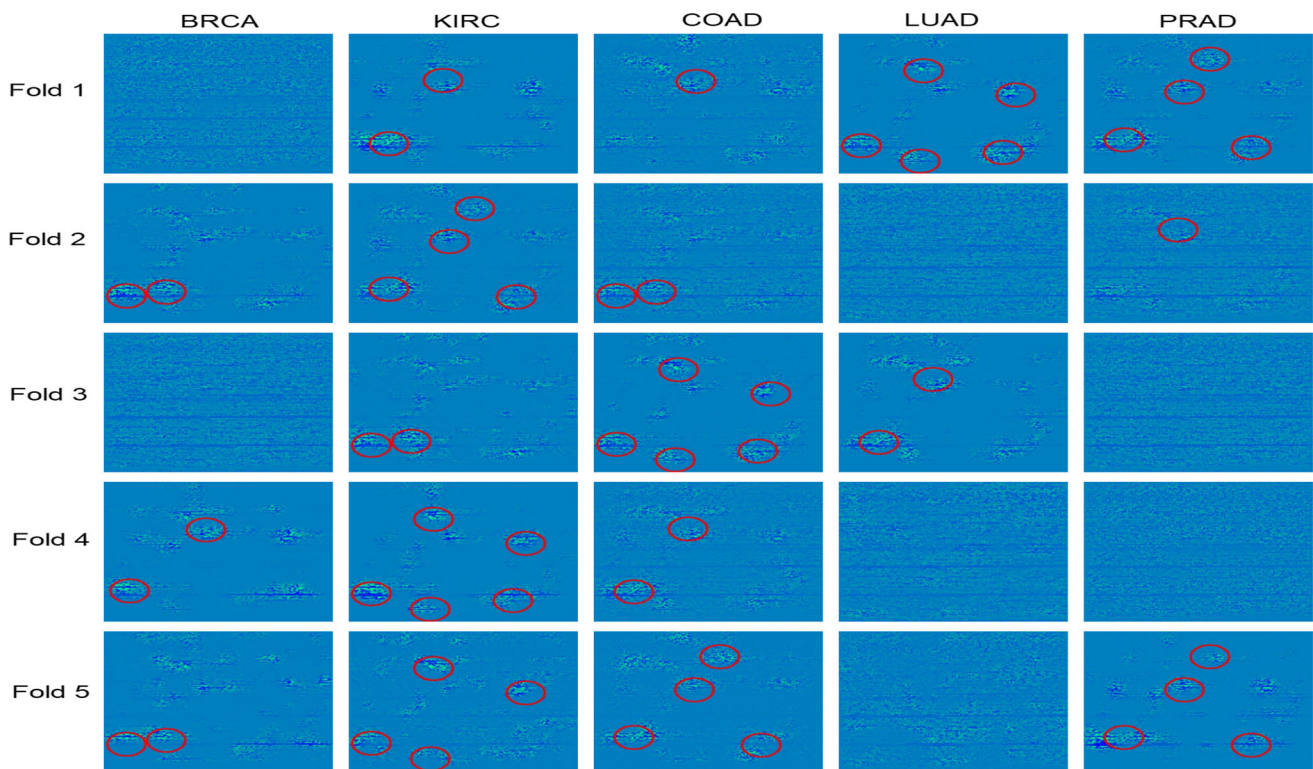


Fig. 8 Confusion matrices of the ensemble classifier



**Fig. 9** Heat map examples for selected cancer types. Each column represents the result from one fold. Rows represent the heat maps of BRCA, KIRC, COAD, LUAD, and PRAD cancer types (from top-down)

### 4.3 Performance analysis of the ensemble model

The ensemble classifier shows about 2% and 4% performance boost across classes and overall in the cases of oncogenes and protein-coding gene, respectively. The accuracy looks easy enough but makes no distinction between classes, e.g., when a doctor makes a medical diagnosis that a patient has cancer, in reality, the patient does not (i.e., false positive). This has very different consequences than making the call that a patient does not have cancer when he actually does (i.e., false negative). A more detailed breakdown of correct and incorrect classifications for each class using the confusion matrices is shown in Fig. 8. The rows of the matrix correspond to ground truth labels, and the columns represent predictions.

There have been significant accuracy improvements in the case of STAD, HNSC, BLCA, THCA, UCEC, and LUSC tumor types, e.g., for the UCEC and LUSC cancer types, not only the accuracies increased slightly, but also the false positive rates reduced significantly. On the contrary, using oncogenes, classifiers were less confused among tumor types, except for the LGG, HNSC, and THCA tumor types. Despite these improvements, there are still high confusion among LUAD, HNSC, and LGG, which is because features from these tumor samples are highly correlated. We could not find an exact reason for

such behavior, and a more gene-specific CNV analysis is required to confirm such correlations.

### 4.4 Validation of the top biomarkers

We report the validation and feature ranking only for the protein-coding gene-based CNVs only. The reason is that the number of protein-coding genes is much higher than that of oncogenes and finding important biomarkers from this large feature space is more challenging. Figure 9 shows some heat maps generated for each class showing similarities across fivefold and displaying a distinct pattern when comparing different cancer types, where each row represents heat maps of one cancer type and columns represent heat maps across five different folds. Circled regions show the similar pattern in all the folds. Although there are some differences among different folds, some patterns are clearly visible.

Since we already identified top genes for which the change in CNV to have significant impacts, only five tumor types (i.e., BRCA, KIRC, COAD, LUAD, and PRAD) have at least five genes having feature importance of at least 0.5. The related genes in the top 20 can then be viewed as tumor-specific biomarkers, which contribute most toward making the prediction by the neural networks.

**Table 6** Top five genes and their importance using protein-coding gene-based CNVs

Cancer type	Gene symbol	Gene type	Importance
BRCA	PIK3CA	Oncogene	0.77453
	TP53	Protein-coding	0.751256
	MLL3	Protein-coding	0.694733
	TBX3	Oncogene	0.574118
	CDH1	Protein-coding	0.5320342
KIRC	VHL	Oncogene	0.696541
	SETD2	Protein-coding	0.6134569
	BAP1	Protein-coding	0.597562
	PTEN	Oncogene	0.5876541
	GUSB	Oncogene	0.546782
LUAD	EGFR	Oncogene	0.812341
	KRAS	Protein-coding	0.723451
	KEAP1	Oncogene	0.703733
	STK11	Protein-coding	0.693567
	NBPF1	Protein-coding	0.6678923
PRAD	SPOP	Oncogene	0.6785425
	TP53	Oncogene	0.5864233
	FOXA1	Protein-coding	0.530765
	COL5A1	Protein-coding	0.519537
	MED12	Protein-coding	0.497568
COAD	ART5	Protein-coding	0.798543
	EPHA6	Protein-coding	0.7543273
	FOXD1	Protein-coding	0.726596
	TIMP1	Protein-coding	0.6954322
	ARID1A	Protein-coding	0.5954237

Then, top five biomarkers and their importance, MAI higher than 0.5, are shown in Table 6. As for the other nine tumor types, only three genes managed to make in the list and are not reported here.

To further validate our findings, the saturation analysis of cancer genes across 13 tumor types (except for COAD) is obtained from the TumorPortal<sup>3</sup> [29]. Validation for the COAD cancer is done based on the signature-based approach [54], which was used for predicting the progression of colorectal cancer. However, our approach makes some false identifications, as 21 out of 25 genes are validated to be correct.

#### 4.5 Analysis of the common biomarkers

We report the top ten common biomarkers while extracting the feature importance in Fig. 10 across 14 cancer types. As seen, KRTAP1-1, INPP5K, GAS8, MC1R, POLR2A,

BET1P1, NAT2, PSD3, KAT6A, and INTS10 genes are common across all the cancer types giving INTS10, a protein-coding gene, the highest feature importance of 0.6.

Importantly, identifying all significant common gene sets for a specific cancer type, e.g., BRCA, will help to understand various aspects of BRCA carcinogenesis. Thus, these top genes have very close relations to the corresponding tumor types, which could be viewed as potential biomarkers. However, a more detailed analysis of biological signaling pathways is needed to further validate these findings.

#### 4.6 Comparisons with related works

Two previous studies focused on using CNVs datasets [13, 53] from cBioPortal for Cancer Genomics. They applied incremental feature selection methods to extract CNV features to train classic ML models and managed to achieve good classification accuracy, i.e., 75% and 85% by Zhang et al. and Sanaa et al., respectively.

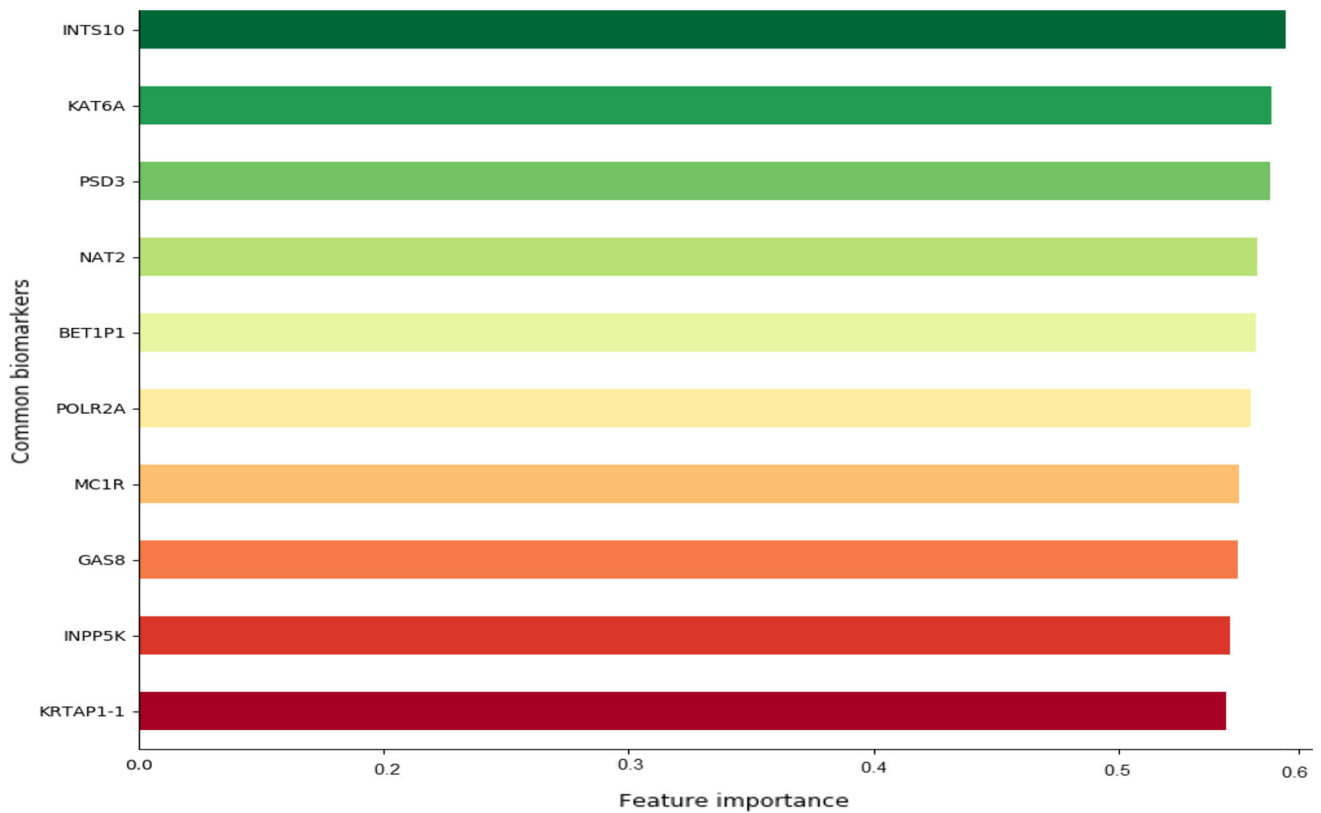
Since our datasets are collected from TCGA and have more samples that are used to train DL algorithms, a one-to-one comparison was not viable. Our proposed approach achieved 80% classification accuracy, which is about 6% better than our previous approach [22] in terms of every metric. Nevertheless, we attempted to provide more human-interpretable interpretation by identifying statistically significance biomarkers. These analyses are further validated with scientific literature, which confirms that the identified genes are biologically relevant and significant CN changes to these gene's location may have significant impact on cancer patients.

## 5 Conclusion

In this paper, we proposed a snapshot neural ensemble method for cancer-type prediction based on CNVs data using two deep architectures called Conv-LSTM and CAE. Our ensemble method is based on cosine annealing techniques, which create multiple model snapshots of these networks. Then, we apply the MAE technique to combine the predictive power of these architectures. Experimental results show that CNVs are not only useful for predicting certain types of cancer but also show an obvious association with cancer growth.

Considering an accuracy of 80% for cancer-type prediction, however, we cannot claim that our model's confidence is very high. This is for several reasons, such as lack of training samples, poor training, and data sparsity. The latter reason caused each network to produce high training and validation loss of error optimization during the training phase, giving lower accuracy. Data sparsity

<sup>3</sup> <http://www.tumorportal.org/>.



**Fig. 10** Common genes across 14 cancer types

evolved because, on average, only 422 samples per tumor were used for the training. This is fairly low for such high-dimensional complex data. As a result, models were not trained well enough to identify subtle differences among tumors types.

We lacked enough training samples compared to the complex dataset, as deep architectures usually expect more samples to get trained well. For the same reason, there was a lack of samples to perform model validation during fivefold cross-validation. Another reason for poor training might be an imbalanced dataset. As we can see, some cancer types, such as BRCA, have 2.5 times more training samples than other tumors. For example, STAD and BLCA have at least 15% sample difference with other tumor types. These two tumors have very low classification accuracy, where about half of the samples were misclassified.

Further, depending on the genome sequencing and CNV calling algorithms, different CNs with different CNV lengths and segmentation means might be produced and the results may change significantly. Although we attempted opening the black-box model through the gene validation and feature ranking, a more detailed analysis on biological signaling pathways is needed to further validate these findings. Since multiple factors are involved in cancer diagnosis, e.g., estrogen receptor (ER), progesterone

receptor (PGR), and human epidermal growth factor receptor 2 (HER2/neu) statuses for breast cancer, providing AI-based diagnoses might not be accurate solely based on CNVs. This requires using multimodal features based on DNA methylation, GE, miRNA expression, and CNVs data by creating a multiplatform network to support each data type.

Lastly, a DL approach using CNV data along with other types of genomics data from different cohorts such as DNA methylation, GE, and somatic mutations will be more reliable. In future, we intend to extend this work by alleviating the above limitations, such as increasing the number of samples for training, testing, and validation, by combining samples from other sources, such as ICGC and COSMIC.

**Acknowledgements** This article is based on a conference paper titled “Cancer Risk and Type Prediction Based on Copy Number Variations with LSTM and Deep Belief Networks” discussed at Artificial Intelligence International Conference (A2IC’2018), in Barcelona, Spain, November 21–23, 2018. We would like to thank anonymous reviewers for their useful comments that helped us to extend and improve the draft and come up with this manuscript.

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give



appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## References

- Ahmad M, Alqarni MA, Khan AM, Hussain R, Mazzara M, Distefano S (2019) Segmented and non-segmented stacked denoising autoencoder for hyperspectral band reduction. *Optik* 180:370–378
- AlShibli A, Mathkour H (2019) A shallow convolutional learning network for classification of cancers based on copy number variations. *Sensors* 19(19):4207
- Blass BE (2017) Editorial for cancer virtual issue
- Buckland PR (2003) Polymorphically duplicated genes: their relevance to phenotypic variation in humans. *Ann Med* 35(5):308–315
- Calcagno DQ et al (2013) MYC, FBXW7 and TP53 copy number variation and expression in gastric cancer. *BMC Gastroenterol* 13(1):141
- Chattopadhyay A, Sarkar A (2018) Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In: Conference on applications of computer vision (WACV), pp 839–847. IEEE
- Chen H et al (2015) Supervised machine learning model for high dimensional gene data in colon cancer detection. *IEEE International congress on big data I*
- Cruz-Roa A et al (2017) Accurate and reproducible invasive breast cancer detection in whole-slide images: a deep learning approach for quantifying tumor extent. *Sci Rep* 7:46450
- Danaee Padideh RG, Hendrix DA (2016) A deep learning approach for cancer detection and relevant gene identification. Pacific symposium on biocomputing Pacific symposium on biocomputing. Vol. 22, NIH Public Access
- David OE, Netanyahu N (2016) Deeppainter: painter classification using deep convolutional autoencoders. In: International conference on artificial neural networks, pp 20–28. Springer
- Ding X, Xue H (2014) Application of machine learning to development of copy number variation-based prediction of cancer risk. *Genomics insights* 7, GEI–S15002
- Diskin SJ, Hou C, Glessner JT, Attiyeh EF, Laudenslager M, Bosse K, Cole K, Mossé YP, Wood A, Lynch JE et al (2009) Copy number variation at 1q21.1 associated with neuroblastoma. *Nature* 459(7249):987
- Elsadek SFA, Makhlof MAA, Aldeen MA (2018) Supervised classification of cancers based on copy number variation. In: International conference on advanced intelligent systems and informatics, pp 198–207. Springer
- Gaul D (2015) Highly-accurate metabolomic detection of early-stage ovarian cancer. *Sci Rep* 5:16351
- Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp 249–256
- Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. *Neural Comput* 18(7):1527–1554
- Hu P, Mitchell H, Li Y, Zhou M, Hazell S (1994) Association of helicobacter pylori with gastric cancer and observations on the detection of this bacterium in gastric cancer cases. *Am J Gastroenterol* 89(10):1806–1810
- Huang G, Li Y, Pleiss G, Liu Z, Hopcroft JE, Weinberger KQ (2017) Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*
- Huang L et al (2011) Copy number variation at 6q13 functions as a long-range regulator and is associated with pancreatic cancer risk. *Carcinogenesis* 33(1):94–100
- Iafraite AJ, Feuk L, Rivera MN, Listewnik ML, Donahoe PK, Qi Y, Scherer SW, Lee C (2004) Detection of large-scale variation in the human genome. *Nat Genet* 36(9):949
- Karim M, Cochez M (2018) Recurrent deep embedding networks for genotype clustering and ethnicity prediction. *arXiv preprint arXiv:1805.12218*
- Karim MR, Beyan O (2018) Cancer risk and type prediction based on copy number variations with LSTM and DBN networks. In: Proceedings of 1st international artificial intelligence conference (A2IC), vol 1. Barcelona, Spain
- Karim MR, Cochez M, Beyan O, Decker S, Lange-Bever C (2018) Onconetexplainer: explainable predictions of cancer types based on gene expression data. *arXiv:1805.07039*
- Karim MR, Wicaksono G, Costa IG, Decker S, Beyan O (2019) Prognostically relevant subtypes and survival prediction for breast cancer based on multimodal genomics data. *IEEE Access* 7, 1–15
- Kingma DP, Salimans T, Welling M (2015) Variational dropout and the local reparameterization trick. In: Advances in neural information processing systems, pp 2575–2583
- Kourou K et al (2015) Machine learning applications in cancer prognosis and prediction. *Comput Struct Biotechnol J* 13:8–17
- Kumaran M, Cass CE, Graham K, Mackey JR, Hubaux R, Lam W, Yasui Y, Damaraju S (2017) Germline copy number variations are associated with breast cancer risk and prognosis. *Sci Rep* 7(1):14621
- Kuusisto KM et al (2013) Copy number variation analysis in familial BRCA1/2-negative Finnish breast and ovarian cancer. *PLoS ONE* 8(8):e71802
- Lawrence MS, Stojanov P, Mermel CH, Robinson JT, Garraway LA, Golub TR, Meyerson M, Gabriel SB, Lander ES, Getz G (2014) Discovery and saturation analysis of cancer genes across 21 tumour types. *Nature* 505(7484):495
- Loshchilov I, Hutter F (2016) Sgdr: Stochastic gradient descent with warm restarts. *arXiv:1608.03983*
- Lyu B, Haque A (2018) Deep learning based tumor type classification using gene expression data. In: Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics, pp 89–96. ACM
- Malekpour SA (2018) Mseq-cnv: accurate detection of copy number variation from sequencing of multiple samples. *Sci Rep* 8(1):4009
- Mamlouk S, Childs LH, Aust D, Heim D, Melching F, Oliveira C, Wolf T, Durek P, Schumacher D, Bläker H et al (2017) DNA copy number changes define spatial patterns of heterogeneity in colorectal cancer. *Nat Commun* 8:14093
- McCarroll SA et al (2006) Common deletion polymorphisms in the human genome. *Nat Genet* 38(1):86–92
- McCarroll SA, Kuruvilla FG, Kirby A (2008) Integrated detection and population-genetic analysis of SNPs and copy number variation. *Nat Genet* 40(10):1166
- Mostavi M, Chiu YC, Huang Y, Chen Y (2019) Convolutional neural network models for cancer type prediction based on gene expression. *arXiv:1906.07794*
- Nguyen DQ, Webber C, Ponting CP (2006) Bias of selection on human copy-number variants. *PLoS Genet* 2(2):e20
- Nie W, Zhang Y, Patel A (2018) A theoretical explanation for perplexing behaviors of backpropagation-based visualizations. *arXiv preprint arXiv:1805.07039*
- Ostrovskaya I, Olshen AB (2010) A classification model for distinguishing copy number variants from cancer-related alterations. *BMC Bioinform* 11(1):297



40. Park RW et al (2015) Identification of rare germline copy number variations over-represented in five human cancer types. *Mol Cancer* 14(1):25
41. Paroder V, Spencer SR, Paroder M, Arango D, Schwartz S, Mariadason JM, Augenlicht LH, Eskandari S, Carrasco N (2006) Na<sup>+</sup>/monocarboxylate transport (SMCT) protein expression correlates with survival in colon cancer: molecular characterization of SMCT. *Proc Nat Acad Sci* 103(19):7270–7275
42. Podolsky MD et al (2016) Evaluation of machine learning algorithm utilization for lung cancer classification based on gene expression levels. *Asian Pac J Cancer Prev* 17(2):835–838
43. Rajanna AR et al (2016) Prostate cancer detection using photoacoustic imaging and deep learning. *Electron Imaging* 2016(15):1–6
44. Selvaraju RR, 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
45. Tomczak Katarzyna PC, Wiznerowicz M (2015) The cancer genome atlas (TCGA): an immeasurable source of knowledge. *Contemp Oncol* 19(1A):A68
46. Torre LA et al (2015) Global cancer statistics 2012. *CA Cancer J Clin* 65(2):87–108
47. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: *Proceedings of the 25th international conference on machine learning*, pp 1096–1103. ACM
48. Willis J, Mukherjee S, Orlov I, Viale A, Offit K, Kurtz RC, Olson S, Klein R (2014) Genome-wide analysis of the role of copy-number variation in pancreatic cancer risk. *Front Genet* 5:29
49. Xingjian S, Chen Z, Wang H, Yeung DY, Wong WK, Woo Wc (2015) Conv-LSTM network: a machine learning approach for precipitation nowcasting. In: *Advances in neural information processing systems*, pp 802–810
50. Yang TL et al (2008) Genome-wide copy-number-variation study identified a susceptibility gene, UGT2B17, for osteoporosis. *Am J Human Genet* 83(6):663–674
51. Yuan Y, Shi Y, Su X, Zou X, Luo Q, Feng DD, Cai W, Han ZG (2018) Cancer type prediction based on copy number aberration and chromatin 3D structure with convolutional neural networks. *BMC Genom* 19(6):97
52. Zhang J, Feuk L, Duggan G, Khaja R, Scherer S (2006) Development of bioinformatics resources for display and analysis of copy number and other structural variants in the human genome. *Cytogenet Genome Res* 115(3–4):205–214
53. Zhang N, Wang M, Zhang P, Huang T (2016) Classification of cancers based on copy number variation landscapes. *Biochim Biophys Acta* 1860(11):2750–2755
54. Zuo S, Dai G, Ren X (2019) Identification of a 6-gene signature predicting prognosis for colorectal cancer. *Cancer Cell Int* 19(1):6

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.