

A APPENDIX A: DEFINITIONS

Below we define the components of a convolutional neural network.

Definition A.1. We define the ReLu function σ , and the softmax function $\bar{\sigma}$ as follows. Here $\underline{x} = (x_1, \dots, x_d)$ for some d .

$$\sigma(\underline{x})_i = \max(x_i, 0); \quad \bar{\sigma}(x)_i = \frac{e^{x_i}}{\sum_{i=1}^k e^{x_i}} \quad \blacksquare$$

Definition A.2. A fully connected layer with n_1 input neurons and n_2 output neurons consists of matrix $A \in \text{Mat}_{n_1, n_2}(\mathbb{R})$ and biases $B \in \mathbb{R}^{n_2}$; we refer to the pair $W = (A, B)$ as the weights of the layer. We define the map $\bar{\phi}_W : \mathbb{R}^{n_1} \rightarrow \mathbb{R}^{n_2}$ as follows (here $v \in \mathbb{R}^{n_1}$).

$$\bar{\phi}_W(v) = \sigma(Av + B) \quad \blacksquare$$

Definition A.3. A *convolutional filter* is a $k \times l$ matrix $w \in \text{Mat}_{k, l}(\mathbb{R})$, which induces the following map. Here $\underline{x} \in \text{Mat}_{m, n}(\mathbb{R})$ and $1 \leq m' \leq m - k + 1, 1 \leq n' \leq n - k + 1$.

$$\begin{aligned} \phi_w : \text{Mat}_{m, n}(\mathbb{R}) &\rightarrow \text{Mat}_{m-k+1, n-k+1}(\mathbb{R}) \\ \phi_w(\underline{x})_{m', n'} &= \sum_{1 \leq k' \leq k, 1 \leq l' \leq l} w_{k', l'} \underline{x}_{k'+m'-1, l'+n'-1} \quad \blacksquare \end{aligned}$$

Definition A.4. A *convolutional layer* consists of a set of convolutional filters $\underline{w} = (w_1, \dots, w_f)$ and biases $\underline{b} = (b_1, \dots, b_f)$. Here $w_i \in \text{Mat}_{k \times l}(\mathbb{R})$ and $b_i \in \mathbb{R}$ for $1 \leq i \leq f$; we refer to the pair $(\underline{w}, \underline{b})$ as the weights of the convolutional layer. The induced map is as follows.

$$\begin{aligned} \bar{\phi}_{\underline{w}}^c : \text{Mat}_{m \times n}(\mathbb{R}) &\rightarrow \text{Mat}_{m-k+1 \times n-k+1}(\mathbb{R})^{\oplus f} \\ \bar{\phi}_{\underline{w}}^c(\underline{x}) &= (\sigma(\phi_{w_1}(\underline{x}) + b_1), \dots, \sigma(\phi_{w_f}(\underline{x}) + b_f)) \end{aligned}$$

In the sum $\phi_{w_i}(\underline{x}) + b_i$, the bias term b_i is added to each co-ordinate of the matrix $\phi_{w_i}(\underline{x})$. \blacksquare

Definition A.5. A *flattening layer* is a linear isomorphism as follows, given by identifying $\text{Mat}_{m', n'}(\mathbb{R})$ with $\mathbb{R}^{m' n'}$.

$$\phi_{fl} : \text{Mat}_{m', n'}(\mathbb{R})^{\oplus f} \rightarrow \mathbb{R}^{m' n' f} \quad \blacksquare$$

B APPENDIX B: PROOFS

B.1 PROOF OF PROPOSITION 1

We start with the proof of Lemma 5.1, which shows how piecewise linear functions can extract features.

Proof of Lemma 5.1. The inequality can be deduced as follows. Here $\underline{x} \in \mathcal{X}_{m, n}$.

$$\begin{aligned} \phi_T(\underline{x}) &> 0 \Leftrightarrow \\ \underline{t}(\underline{x}_{[i+1, i+k], [j+1, j+l]}) &< \epsilon \text{ for some } (i, j) \in \mathcal{R}_{k, l}^{m, n} \\ \Leftrightarrow \underline{x}_{[i+1, i+k], [j+1, j+l]} &\in \mathcal{X}^T \text{ for some } (i, j) \in \mathcal{R}_{k, l}^{m, n} \end{aligned}$$

By definition, this is equivalent to saying that $\underline{x} \in \mathcal{X}_{m, n}^T$. \square

Proof of Proposition 1. This can be deduced from Lemma 4.3 using the following argument. Note that since $\phi_{T_i}(\underline{x}) \geq 0$,

$$\phi_I(\underline{x}) = \sum_{1 \leq i \leq q} \phi_{T_i}(\underline{x}) > 0$$

if and only if $\phi_{T_i}(\underline{x}) > 0$ for some i . This is true precisely when $\underline{x} \in \mathcal{X}^{T_i}$ for some i . In other words, it is true precisely when $\underline{x} \in \mathcal{X}_{m, n}^T$. \square

B.2 PROOF OF THEOREM 1

We start a constructive proof of Lemma 5.2, which will play a key role in the proof of Theorem 1.

Proof of Lemma 5.2. Recall the definition of $\phi_T(\underline{x})$, as follows.

$$\phi_T(\underline{x}) = \sum_{(i,j) \in \mathcal{R}_{k,l}^{m,n}} \max(0, \epsilon - \underline{t}(\underline{x}_{[i+1,i+k],[j+1,j+l]}))$$

To simplify the notation, given $(i, j) \in \mathcal{R}_{k,l}^{m,n}$, define the following quantity.

$$\phi_{(i,j)}(\underline{x}) = \max(0, \epsilon - \underline{t}(\underline{x}_{[i+1,i+k],[j+1,j+l]}))$$

First we show that there exists a neural network $\mathcal{N}'[T]$ such that the following holds.

$$f_{\mathcal{N}'[T]} = [\phi_{(i,j)}(\underline{x})]_{(i,j) \in \mathcal{R}_{k,l}^{m,n}}$$

Define the four binary matrices as follows.

$$\begin{aligned} w_{11} &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, & w_{12} &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \\ w_{21} &= \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, & w_{22} &= \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \end{aligned}$$

Define the vectors $\underline{w}_1 \in \text{Mat}_2(\mathbb{R})^4$ and $\underline{b}_0, \underline{b}_1 \in \mathbb{R}^4$ as follows.

$$\begin{aligned} \underline{w}_1 &= (w_{11}, w_{12}, w_{21}, w_{22}) \\ \underline{b}_0 &= (0, 0, 0, 0); & \underline{b}_1 &= (1, 1, 1, 1) \end{aligned}$$

Define S , the set of all non-zero entries in the matrix \underline{t} , as follows.

$$S = \{t_{u,v} \mid (u, v) \in \text{supp}(\underline{t})\}$$

Let $d = |S|$. For convenience, we relabel the entries, so that $S = \{s_1, \dots, s_d\}$.

We specify the weights $(\underline{w}, \underline{b})$ of the convolutional layer below. Here $\underline{w} \in \text{Mat}_{2,2}(\mathbb{R})^{4(d+1)}$ denotes the convolutional filters, and $\underline{b} \in \mathbb{R}^{4(d+1)}$ denotes the biases.

$$\begin{aligned} \underline{w} &= (\underline{w}_1, 2\underline{w}_1, \dots, 2\underline{w}_1) \\ \underline{b} &= (\underline{b}_0, -2s_1\underline{b}_1, \dots, -2s_d\underline{b}_1) \end{aligned}$$

Let \underline{x}' be the image of the input image \underline{x} under the map $\phi_{(\underline{w}, \underline{b})}$ induced by the convolutional layer composed with a flattening layer.

$$\underline{x}' = \phi_f(\phi_{(\underline{w}, \underline{b})}\underline{x}) \in \mathbb{R}^{4(d+1)(m-1)(n-1)}$$

By the above construction, it is clear that the quantities $\sigma(x_{i',j'})$ and $\sigma(2x_{i',j'} - 2s_l)$ appear as coordinates of \underline{x}' (for all $1 \leq i' \leq m, 1 \leq j' \leq n$ and $1 \leq l \leq d$). We will need the following identity to obtain the quantities $\|x_{i',j'} - s_l\|$.

$$\begin{aligned} \|y - c\| &= \max(y - c, c - y) \\ &= \max(2y - 2c, 0) + (c - y) \\ &= \sigma(2y - 2c) - \sigma(y) + c \end{aligned}$$

Using the above equation and the definition, $\underline{t}(\underline{x}_{[i+1,i+k],[j+1,j+l]})$ can be expressed as a linear combination of the quantities $\sigma(x_{i',j'})$ and $\sigma(2x_{i',j'} - 2s_l)$, with a constant term. It follows that there exists a fully connected layer $\phi_{(\underline{w}', \underline{b}')}$ with the following property.

$$\phi_{(\underline{w}', \underline{b}')} \underline{x}' = [\phi_{(i,j)}(\underline{x})]_{(i,j) \in \mathcal{R}_{k,l}^{m,n}}$$

The desired neural network $\mathcal{N}[T]$ can be obtained by adding a fully connected layer with one output neuron to the network $\mathcal{N}'[T]$, with all weights equal to 1. The resulting network $\mathcal{N}(T)$ has $4(|\text{supp}(\underline{t})| + 1)$ convolutional filters with dimension 2×2 . The fully connected layer has $|\mathcal{R}_{k,l}^{m,n}|$ neurons, and $|\mathcal{R}_{k,l}^{m,n}| < mn$. The conclusion follows. \square

Next we extend Lemma 4.4, from functions corresponding to feature tiles to the analogous functions for images.

Lemma 4.5 *Let \mathcal{I} be an image, consisting of feature tiles $\{T_1, \dots, T_q\}$. There exists a neural network $\mathcal{N}[\mathcal{I}]$ with one convolutional layer and one fully connected layer such that the following holds.*

$$f_{\mathcal{N}[\mathcal{I}]}(\underline{x}) = \phi_{\mathcal{I}}(\underline{x})$$

Proof of Lemma 4.5. We construct the networks $\mathcal{N}(T_1), \dots, \mathcal{N}(T_q)$ from the above Lemma. Recalling the definition of $\phi_{\mathcal{F}}(\underline{x})$, it suffices to construct a network $\mathcal{N}(\mathcal{I})$ with the following property.

$$f_{\mathcal{N}(\mathcal{I})}(\underline{x}) = f_{\mathcal{N}(T_1)}(\underline{x}) + \dots + f_{\mathcal{N}(T_q)}(\underline{x})$$

The convolutional (resp. fully connected) layer of $\mathcal{N}[\mathcal{I}]$ is obtained by concatenating the convolutional (resp. fully connected) layers of $\mathcal{N}(T_i)$, for $1 \leq i \leq q$. The weights are chosen so that the above identity holds. \square

Theorem 1 now follows from Lemma 4.5 and Proposition 1, as described below.

Proof of Theorem 1. For each image \mathcal{I}_j , denote the constituent feature tiles as follows. $\mathcal{I}_j = \{\mathcal{T}_1^j, \dots, \mathcal{T}_{r_j}^j\}$. By Lemma 4.6 and Definition 4.4, there exists networks \mathcal{N}' and \mathcal{N}'' such that the following holds.

$$\begin{aligned} f_{\mathcal{N}'}(\underline{x}) &= [\phi_{\mathcal{T}_1^1}(\underline{x}), \dots, \phi_{\mathcal{T}_{r_1}^1}(\underline{x}), \dots, \phi_{\mathcal{T}_1^{r_1}}(\underline{x}), \dots, \phi_{\mathcal{T}_{r_{r_1}}^{r_1}}(\underline{x})] \\ f_{\mathcal{N}''}[\phi_{\mathcal{T}_1^1}(\underline{x}), \dots, \phi_{\mathcal{T}_{r_1}^1}(\underline{x}), \dots, \phi_{\mathcal{T}_1^{r_1}}(\underline{x}), \dots, \phi_{\mathcal{T}_{r_{r_1}}^{r_1}}(\underline{x})] &= [\phi_{\mathcal{I}_1}(\underline{x}), \dots, \phi_{\mathcal{I}_{r_1}}(\underline{x})] \end{aligned}$$

By composing the two networks \mathcal{N}' and \mathcal{N}'' , and adding a softmax layer at the end, we obtain the desired network $\mathcal{N}[\mathcal{I}]$, which has one convolutional layer and one fully connected layers. \square

B.3 PROOF OF THEOREM 2

First we will prove Lemma 5.3; we start with two preparatory Lemmas.

Lemma B.1. Given $a \in \mathbb{R}$, there exists a convolutional neural network \mathcal{A} with two convolutional layers that has the following property. Below $D_a(\underline{x})$ denotes the matrix from the above definition, with $a \in \text{Mat}_{1,1}(\mathbb{R})$.

$$f_{\mathcal{A}}(\underline{x}) = D_a(\underline{x})$$

Both convolutional layers have 1×1 kernels; the former has two filters, and the latter has one filter.

Proof. We use the following identity.

$$\|x_{i,j} - a\| = \max(x_{i,j} - a, a - x_{i,j}) = \sigma(2x_{i,j} - 2a) - \sigma(x_{i,j}) + a \quad \text{for } y, c \in \mathbb{R}$$

The first convolutional layer has two filters, and its weights and biases are chosen so that the corresponding outputs are $\sigma(2x_{i,j} - 2a)$ and $\sigma(x_{i,j})$. The second convolutional layer has one filter, and its weights are chosen so that the output is $\|x_{i,j} - a\|$ (using the above identity). \square

Lemma B.2. Let $t \in \text{Mat}_{2k,2k}(\mathbb{R})$ and $\underline{x} \in \mathcal{X}_{m,n}$ be matrices (as in Definition 3.2-3.4). We divide t into four smaller matrices as follows.

$$t_{11} = t_{1:k,1:k}; t_{12} = t_{1:k,k+1:2k}$$

$$t_{21} = t_{k+1:2k,1:k}; t_{22} = t_{k+1:2k,k+1:2k}$$

There exists a convolutional layer with weights $\underline{w}(t)$, satisfying the following property.

$$\phi_{\underline{w}(t)}^c(D_{t_{11}}(\underline{x}), D_{t_{12}}(\underline{x}), D_{t_{21}}(\underline{x}), D_{t_{22}}(\underline{x})) = D_t(\underline{x})$$

The convolutional layer has one filter and $k \times k$ kernels.

Proof. We use the following identity, which follows from the definitions.

$$\begin{aligned} \underline{t}(\underline{x}_{i+1:i+2n,j+1:j+2n}) &= \underline{t}_{11}(\underline{x}_{i+1:i+n,j+1:j+n}) + \underline{t}_{12}(\underline{x}_{i+1:i+n,j+n+1:j+2n}) \\ &\quad + \underline{t}_{21}(\underline{x}_{i+n+1:i+2n,j+1:j+n}) + \underline{t}_{22}(\underline{x}_{i+n+1:i+2n,j+n+1:j+2n}) \end{aligned}$$

Let $\underline{w}(t) = (w_1, w_2, w_3, w_4)$, with the matrices $w_1, w_2, w_3, w_4 \in \text{Mat}_{k,k}(\mathbb{R})$ defined below (here $E_{i,j} \in \text{Mat}_{k,k}(\mathbb{R})$ denotes a matrix with a 1 in the (i, j) -th position, and zeroes elsewhere).

$$w_1 = E_{11}, w_2 = E_{1,k}, w_3 = E_{k,1}, w_4 = E_{k,k}$$

From the above expression for $\underline{t}(\underline{x}_{i+1:i+2n,j+1:j+2n})$, it follows that the map $\phi_{\underline{w}(t)}^c$ has the desired property. \square

Proof of Lemma 5.3. We proceed by induction. The $r = 1$ case can be deduced from Lemma B.1 as follows. As in Lemma B.1, we construct the first convolutional layer so that the outputs are $\sigma(2x_{i,j} - 2a)$ and $\sigma(x_{i,j})$. We choose the weights of the second convolutional layer so that the resulting output is $D_t(\underline{x})$, by using the identities in Lemma B.1 and Lemma B.2.

For the inductive step, we argue as follows. We divide t into four smaller matrices - t_{11}, t_{12}, t_{21} and t_{22} - as in Lemma B.2. By the inductive hypothesis, there exists convolutional neural networks $\mathcal{N}[t_{ij}]$ such that $f_{\mathcal{N}[t_{ij}]}(\underline{x}) = D_{t_{ij}}(\underline{x})$ (here $1 \leq i, j \leq 2$). By concatenating these four networks and adding the layer $\phi_{\underline{w}(t)}$ from Lemma B.2, we obtain the desired convolutional network whose output is $D_t(\underline{x})$. \square

We introduce a definition of padded tiles, and then proceed to construct convolutional neural networks that express the piecewise linear functions $\phi_{\mathcal{I}_j}(\underline{x})$, proving Theorem 2.

Definition B.1. Let $T = (t, \epsilon)$ be a feature tile with dimension $k \times l$ where $k, l < 2^r$. Define the enlarged tile $T^{(r)} = (t^{(r)}, \epsilon)$ to be the feature tile where $t^{(r)} \in \text{Mat}_{2^r, 2^r}(\mathbb{R})$ is obtained by padding the matrix $t \in \text{Mat}_{k,l}(\mathbb{R})$ with zeroes (so that $t_{1:k,1:l}^{(r)} = t$). Given an image $\mathcal{F} = \{T_1, \dots, T_q\}$, define the enlarged image $\mathcal{F}^{(r)}$ as follows:

$$\mathcal{F}^{(r)} = \{T_1^{(r)}, \dots, T_q^{(r)}\}$$

Proof of Theorem 2. Our convolutional neural network will consist of a padding layer p , $r + 1$ convolutional layers, and two fully connected layers. We start with a padding layer p which adds 2^r pixels on each of the four sides of the input.

For each image class \mathcal{I}_j , denote by $\{t_1^j, \dots, t_{r_j}^j\}$ the tiles appearing in the features that constitute \mathcal{I}^j . Using Lemma 4.9, there exists a convolutional neural networks \mathcal{N}' such that the following holds.

$$f_{\mathcal{N}'}(\underline{x}) = [D_{t_1^{1(r)}}(p(\underline{x})), \dots, D_{t_{r_1}^{1(r)}}(p(\underline{x})), \dots, D_{t_1^{l(r)}}(p(\underline{x})), \dots, D_{t_{r_l}^{l(r)}}(p(\underline{x}))]$$

From Definition 4.6, it is easy to see that there exists a fully connected network \mathcal{N}'' with two layers such that the following holds.

$$\begin{aligned} f_{\mathcal{N}''}[D_{t_1^{1(r)}}(p(\underline{x})), \dots, D_{t_{r_1}^{1(r)}}(p(\underline{x})), \dots, D_{t_1^{l(r)}}(p(\underline{x})), \dots, D_{t_{r_l}^{l(r)}}(p(\underline{x}))] &= [\phi_{\mathcal{I}_1^{(r)}}(p(\underline{x})), \dots, \phi_{\mathcal{I}_l^{(r)}}(p(\underline{x}))] \\ &= [\phi_{\mathcal{I}_1}(\underline{x}), \dots, \phi_{\mathcal{I}_l}(\underline{x})] \end{aligned}$$

By composing the two networks \mathcal{N}' and \mathcal{N}'' , and adding a softmax layer at the end, we obtain the desired network $\mathcal{N}[\mathcal{I}]$, which has r convolutional layers and 2 fully connected layers. \square