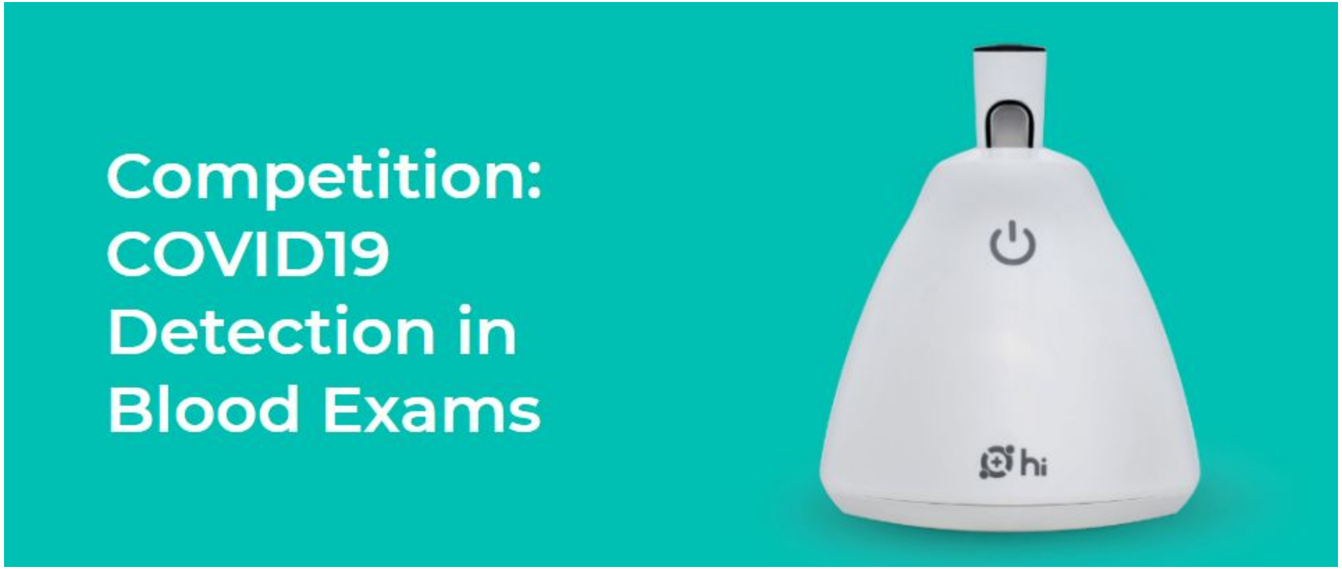
IJCNN 2021- 1st Place Solution

Outline

- Introduction
- Competition Data
- Metric and Loss Function
- Model Selection
- Model
- Results
- Improvements
- References

Introduction

The International Joint
Conference on Neural Networks



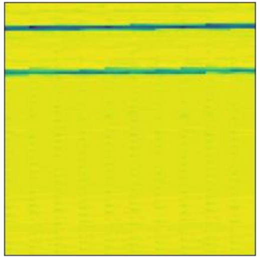
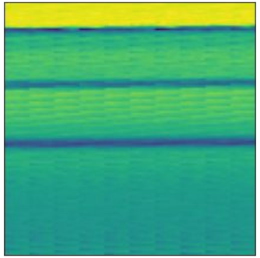
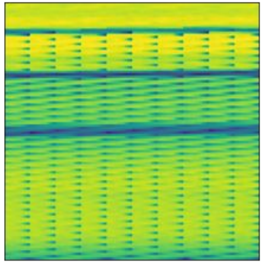
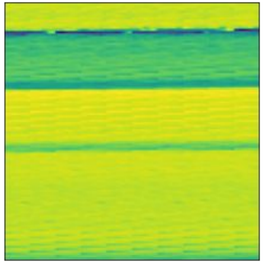
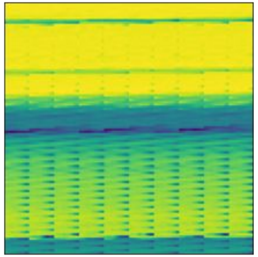
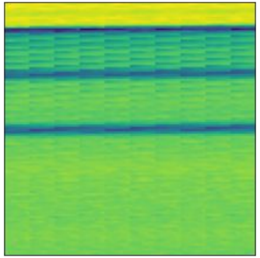
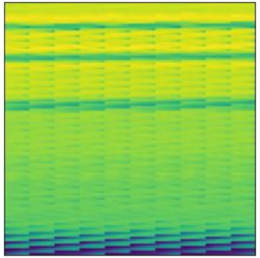
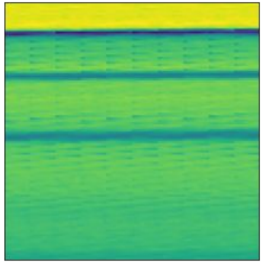
Competition Data

	0	1	2	3	4	5	6	7	8	9	...	12200	12201	12202	12203	12204	12205	12206	12207	12208	12209
0	0.972549	0.972549	0.968627	0.968627	0.968627	0.968627	0.968627	0.968627	0.976471	0.980392	...	0.980392	0.980392	0.980392	0.980392	0.980392	0.980392	0.980392	0.980392	0.980392	0.980392
1	0.984314	0.988235	0.992157	0.988235	0.984314	0.984314	0.984314	0.984314	0.984314	0.980392	...	0.694118	0.690196	0.666667	0.674510	0.662745	0.627451	0.627451	0.607843	0.549020	0.541176
2	0.929412	0.937255	0.941176	0.937255	0.937255	0.941176	0.941176	0.933333	0.949020	0.945098	...	0.823529	0.819608	0.819608	0.807843	0.792157	0.788235	0.780392	0.752941	0.721569	0.705882
3	0.847059	0.847059	0.843137	0.839216	0.835294	0.831373	0.827451	0.827451	0.815686	0.819608	...	0.337255	0.345098	0.368627	0.392157	0.411765	0.431373	0.447059	0.466667	0.486275	0.498039
4	0.964706	0.972549	0.980392	0.980392	0.976471	0.976471	0.976471	0.972549	0.972549	0.976471	...	0.764706	0.776471	0.776471	0.776471	0.776471	0.772549	0.772549	0.768627	0.768627	0.768627
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
40039	0.933333	0.933333	0.937255	0.945098	0.952941	0.952941	0.949020	0.945098	0.941176	0.945098	...	0.937255	0.941176	0.937255	0.921569	0.894118	0.878431	0.847059	0.792157	0.760784	0.760784
40040	0.976471	0.976471	0.976471	0.976471	0.972549	0.968627	0.964706	0.964706	0.968627	0.972549	...	0.756863	0.756863	0.760784	0.768627	0.776471	0.772549	0.776471	0.780392	0.772549	0.776471
40041	0.839216	0.839216	0.839216	0.843137	0.850980	0.858824	0.866667	0.874510	0.874510	0.874510	...	0.721569	0.721569	0.721569	0.721569	0.721569	0.721569	0.721569	0.721569	0.721569	0.725490
40042	0.737255	0.674510	0.662745	0.647059	0.513725	0.392157	0.270588	0.192157	0.105882	0.070588	...	0.827451	0.827451	0.827451	0.827451	0.827451	0.827451	0.827451	0.823529	0.823529	0.823529
40043	0.901961	0.901961	0.898039	0.898039	0.898039	0.898039	0.898039	0.898039	0.898039	0.894118	...	0.847059	0.847059	0.847059	0.850980	0.850980	0.850980	0.854902	0.854902	0.854902	0.862745

40044 rows × 12210 columns

Competition Data

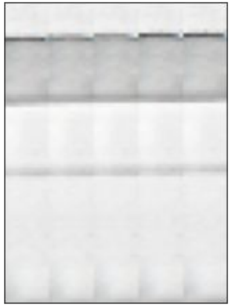
Reshape: 12210 -> (111, 110)

Competition Data

Reshape: 12210 -> (74, 55, 3)








Metric and Loss Function

- Competition Metric: Macro F1-Score
- As Dataset is well balanced (50 - 50), we can use Binary Cross Entropy. No need for custom loss function.

$$Macro\ F_1 = \frac{1}{C}\sum_{c=1}^{C}\frac{2 \cdot TPR_c \cdot PPV_c}{TPR_c + PPV_c} \tag{1}$$

$$TPR_c = TP_c/(TP_c + FN_c) \tag{2}$$

$$PPV_c = TP_c/(TP_c + FP_c) \tag{3}$$

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Model Selection

- Ideally 10-Fold cross validation, but it’s computationally expensive
- Train-test split 75/25

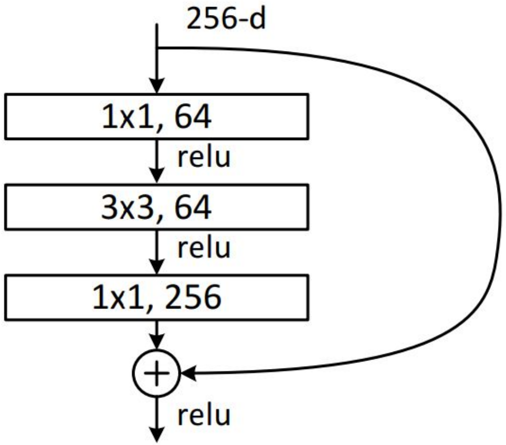
Model

- Residual SE Block
  - Bottleneck Residual Block
  - Full Pre-Activation
  - Squeeze and Excite Attention
  - Stochastic Depth
- EfficientNet Scaling

```
model = Sequential((

    Input((74, 55, 3)),

    Conv2D(128, 3),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),

    ResidualSEBlock(128, strides=2),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),

    ResidualSEBlock(128, strides=2),
    ResidualSEBlock(128, survival_prob=0.7),
    ResidualSEBlock(256, survival_prob=0.7),
    ResidualSEBlock(256, survival_prob=0.7),

    ResidualSEBlock(256, strides=2),
    ResidualSEBlock(256, survival_prob=0.6),
    ResidualSEBlock(512, survival_prob=0.6),
    ResidualSEBlock(512, survival_prob=0.6),

    ResidualSEBlock(512, strides=2),
    ResidualSEBlock(512, survival_prob=0.5),
    ResidualSEBlock(512, survival_prob=0.5),
    ResidualSEBlock(512, survival_prob=0.5),

    BatchNormalization(),
    Activation('swish'),
    GlobalAveragePooling2D(),
    Dense(1, activation='sigmoid')

))
```

Residual SE Block

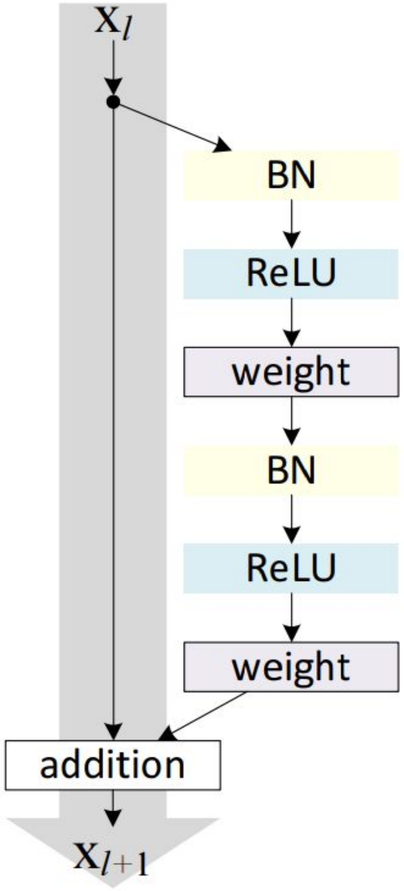
- **Bottleneck Residual Block**
- Full Pre-Activation
- Squeeze and Excite Attention
- Stochastic Depth



(He, 2015)

Residual SE Block

- Bottleneck Residual Block
- **Full Pre-Activation**
  - Batch Normalization
  - SiLU Activation
- Squeeze and Excite Attention
- Stochastic Depth



(He, 2016)

Residual SE Block

- Bottleneck Residual Block
- **Full Pre-Activation**
    - **Batch Normalization**
    - SiLU Activation
- Squeeze and Excite Attention
- Stochastic Depth

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

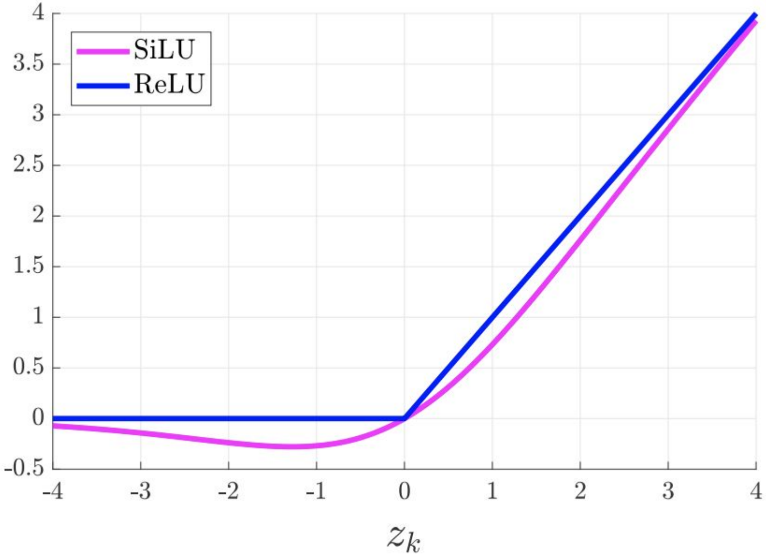
Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

(Ioffe, 2015)

Residual SE Block

- Bottleneck Residual Block
- **Full Pre-Activation**
  - Batch Normalization
  - **SiLU Activation**
- Squeeze and Excite Attention
- Stochastic Depth

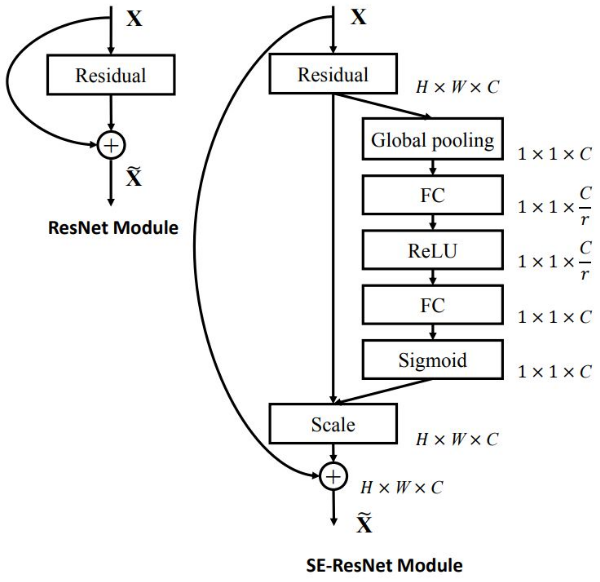
$$SiLU(x) = x * \sigma(x) = x * \frac{1}{1 + e^{-x}}$$

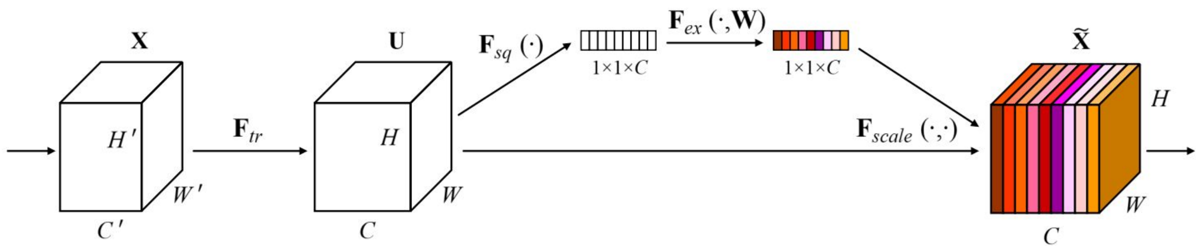


(Elfwing, 2017)

Residual SE Block

- Bottleneck Residual Block
- Full Pre-Activation
- **Squeeze and Excite Attention**
- Stochastic Depth

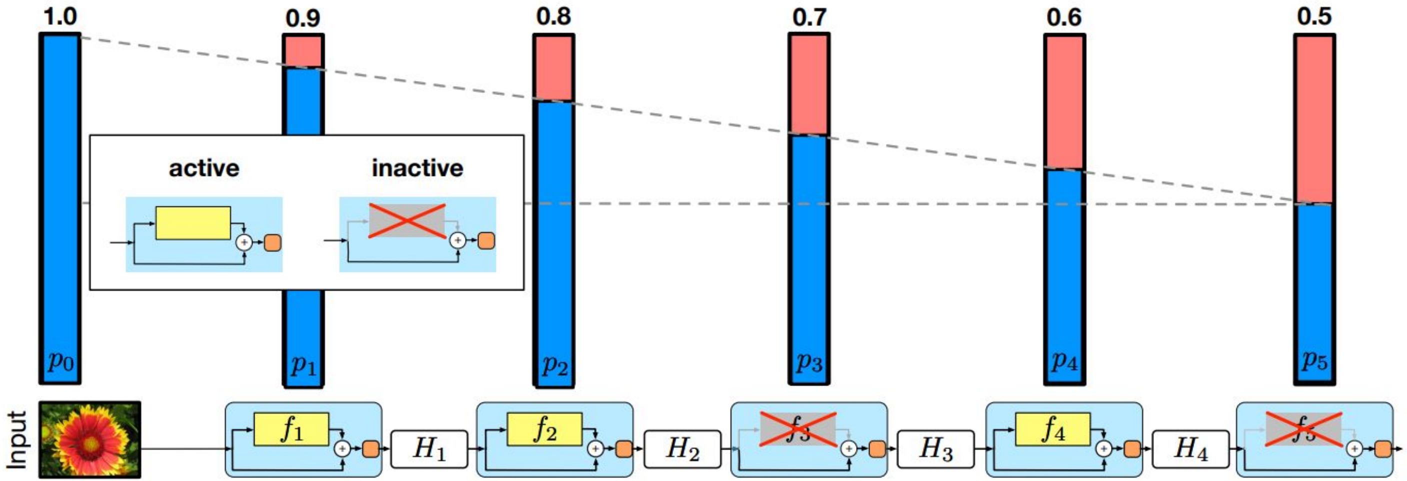




(Hu, 2019)

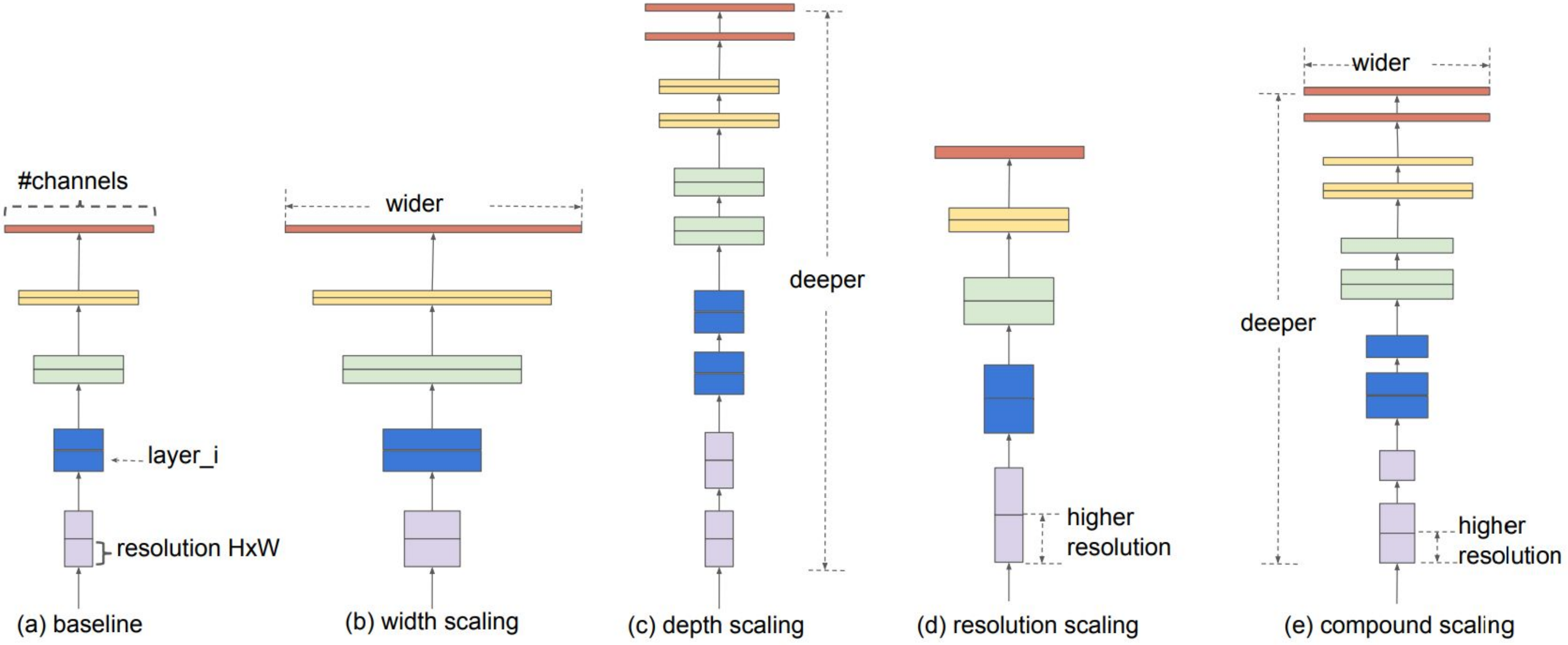
Residual SE Block

- Bottleneck Residual Block
- Full Pre-Activation
- Squeeze and Excite Attention
- **Stochastic Depth**



(Huang, 2016)

EfficientNet Scaling



(Tan, 2020)

EfficientNet Scaling

$$
\begin{aligned}
\text{depth: } & d = \alpha^{\phi} \\
\text{width: } & w = \beta^{\phi} \\
\text{resolution: } & r = \gamma^{\phi} \\
\text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\
& \alpha \geq 1, \beta \geq 1, \gamma \geq 1
\end{aligned}
$$

(Tan, 2020)

Model

- Residual SE Block
  - Bottleneck Residual Block
  - Full Pre-Activation
  - Squeeze and Excite Attention
  - Stochastic Depth
- EfficientNet Scaling

```
model = Sequential((

    Input((74, 55, 3)),

    Conv2D(128, 3),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),

    ResidualSEBlock(128, strides=2),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),
    ResidualSEBlock(128, survival_prob=0.8),

    ResidualSEBlock(128, strides=2),
    ResidualSEBlock(128, survival_prob=0.7),
    ResidualSEBlock(256, survival_prob=0.7),
    ResidualSEBlock(256, survival_prob=0.7),

    ResidualSEBlock(256, strides=2),
    ResidualSEBlock(256, survival_prob=0.6),
    ResidualSEBlock(512, survival_prob=0.6),
    ResidualSEBlock(512, survival_prob=0.6),

    ResidualSEBlock(512, strides=2),
    ResidualSEBlock(512, survival_prob=0.5),
    ResidualSEBlock(512, survival_prob=0.5),
    ResidualSEBlock(512, survival_prob=0.5),

    BatchNormalization(),
    Activation('swish'),
    GlobalAveragePooling2D(),
    Dense(1, activation='sigmoid')

))
```

Optimizer and Training

- Adam Optimizer with Learning Rate Decay
    - Initial Learning Rate: 0.001
    - Decay Epochs: 10
    - Decay Rate: 0.1
    - Staircase = True
- Batch Size = 32
- Epochs = 50

(Kingma, 2017)

Results

- Train / Test: 0.92 / 0.91

Team	Macro F1 Score	Position
Alantlb	0.82	1st
Atish Kumar Dipongkor	0.70	2nd
CoV-Unica-Team	0.68	3rd

References

(He, 2015) Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. (2015). Deep Residual Learning for Image Recognition.

(He, 2016) Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. (2016). Identity Mappings in Deep Residual Networks.

(Ioffe, 2015) Sergey Ioffe, & Christian Szegedy. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

(Elfwing, 2017) Stefan Elfwing, Eiji Uchibe, & Kenji Doya. (2017). Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning.

(Hu, 2019) Jie Hu, Li Shen, Samuel Albanie, Gang Sun, & Enhua Wu. (2019). Squeeze-and-Excitation Networks.

(Huang, 2016) Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, & Kilian Weinberger. (2016). Deep Networks with Stochastic Depth.

(Tan, 2020) Mingxing Tan, & Quoc V. Le. (2020). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.

(Kingma, 2017) Diederik P. Kingma, & Jimmy Ba. (2017). Adam: A Method for Stochastic Optimization.