Models inferenced from 2019 paper published by Sir Ansari

```
model = Sequential()
model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D((2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(AveragePooling2D((2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(AveragePooling2D((2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))
model.add(Dense(128, activation= 'relu', use_bias=True))
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()
model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D((2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(AveragePooling2D((2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(AveragePooling2D((2, 2)))
model.add(Dropout(rate=0.25))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))
model.add(Dense(128, activation= 'relu', use_bias=True))
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()
model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))
model.add(Dense(128, activation= 'relu', use_bias=True))
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()
model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))
model.add(Dense(128, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()
model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(128, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(128, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'relu', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
```

```
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(32, activation= 'relu', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(32, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(32, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
```

```
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(32, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(32, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(64, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(3,3), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(64, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(5,5), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(5,5), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(5,5), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(64, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(16, kernel_size=(5,5), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(5,5), activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(5,5), activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.2))
model.add(Dense(64, activation= 'relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(32, kernel_size=(5,5), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(5,5), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(5,5), activation= 'relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))
model.add(Dense(64, activation= 'relu', use_bias=True))
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()
model.add(Conv2D(16, kernel_size=(5,5), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(32, kernel_size=(5,5), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Conv2D(64, kernel_size=(5,5), activation= 'relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))
model.add(Flatten())
model.add(Dense(256, activation= 'relu', use_bias=True))
model.add(Dropout(rate=0.2))
model.add(Dense(64, activation= 'relu', use_bias=True))
```

```
model.add(Dense(2, activation='softmax', use_bias=True))
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Flatten())
model.add(Dense(128, activation='relu', use_bias=True))
model.add(Dropout(rate=0.5))

model.add(Dense(64, activation='relu', use_bias=True))
model.add(Dropout(rate=0.5))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(128, activation='relu', use_bias=True))
model.add(Dropout(rate=0.5))

model.add(Dense(64, activation='relu', use_bias=True))
model.add(Dropout(rate=0.5))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(128, activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))
```

```
model.add(Dense(64, activation='relu', use_bias=True))
model.add(BatchNormalization())

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(128, activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Dense(64, activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Flatten())
model.add(Dense(128, activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Dense(64, activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(16, kernel_size=(5,5), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Conv2D(32, kernel_size=(5,5), activation='relu', use_bias=True))
```

```
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Conv2D(64, kernel_size=(5,5), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(rate=0.25))

model.add(Flatten())
model.add(Dense(128, activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Dense(64, activation='relu', use_bias=True))
model.add(BatchNormalization())
model.add(Dropout(rate=0.5))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Conv2D(32, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Conv2D(64, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Flatten())
model.add(Dense(128, activation='relu', use_bias=True))
model.add(Dropout(rate=0.5))

model.add(Dense(64, activation='relu', use_bias=True))
model.add(Dropout(rate=0.5))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()

model.add(Conv2D(8, kernel_size=(3,3), activation='relu', input_shape=(64,64,3), use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', use_bias=True))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))

model.add(Flatten())
model.add(Dense(32, activation='relu', use_bias=True))
model.add(Dropout(rate=0.5))

model.add(Dense(2, activation='softmax', use_bias=True))

model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])
```

Models inferenced from 2023 paper published by Sir Ansari

```
model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
```

```
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(17, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(18, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(19, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
```

```
model.add(Dropout(rate=0.5))

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(20, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(19, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(16, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(18, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(17, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(23, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(18, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(26, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(17, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)
```

```
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(19, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(16, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(20, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(16, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(19, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))
```

```
model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(18, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='relu', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(18, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(18, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))
```

```
model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(16, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(19, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(16, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(20, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(19, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))
```

```
model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])


model = Sequential()

model.add(Conv2D(14, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(15, kernel_size=(3,3), activation='sigmoid', input_shape = (64,64,3), use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Conv2D(17, kernel_size=(3,3), activation='sigmoid', use_bias=True))
model.add(AveragePooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.5))

model.add(Flatten())
model.add(Dense(256, activation= 'sigmoid', use_bias=True))
model.add(Dropout(rate=0.2))

model.add(Dense(128, activation= 'sigmoid', use_bias=True))

model.add(Dense(2, activation='softmax', use_bias=True))

optimizer = Adam(learning_rate=0.001)

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```