# A Study of Aggregation of Long Time-series Input for LSTM Neural Networks

No Author Given

No Institute Given

## 1 Used Models

**Listing 1.1.** Used Models

```
# Single cell LSTM
model = Sequential()
model.add(LSTM(units=8, activation='relu', name='first_lstm', recurrent_dropout=0.1,
               input_shape=(Xtrain.shape[1], Xtrain.shape[2])))
model.add(Dense(1, activation=LAST_ACTIVATION))

# Stacked LSTM
model = Sequential()
model.add(LSTM(100, activation='relu', return_sequences=True, recurrent_dropout=0.1,
               input_shape=(Xtrain.shape[1], Xtrain.shape[2])))
model.add(LSTM(50, activation='relu', return_sequences=True, recurrent_dropout=0.1))
model.add(LSTM(30, activation='relu', recurrent_dropout=0.2))
model.add(Dense(1, activation=LAST_ACTIVATION))

# Single cell RNN
model = Sequential()
model.add(RNN(units=8, activation='relu', name='first_lstm', recurrent_dropout=0.1,
               input_shape=(Xtrain.shape[1], Xtrain.shape[2])))
model.add(Dense(1, activation=LAST_ACTIVATION))

# Bidirectional LSTM
model = Sequential()
model.add(Bidirectional(LSTM(100, return_sequences=True, activation='relu')))
model.add(Bidirectional(LSTM(50, return_sequences=True, activation='relu')))
model.add(Bidirectional(LSTM(20, activation='relu')))
model.add(Dense(1, activation=LAST_ACTIVATION))

# "CNN":
model = Sequential()
model.add(Conv1D(filters=128, kernel_size=2, activation='relu', name='extractor',
                 input_shape=(Xtrain.shape[1], Xtrain.shape[2])))
model.add(Dropout(0.5))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=2, activation='relu'))
model.add(Dropout(0.5))
```

```
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(50, activation='relu'))
model.add(Dense(1, activation=LAST_ACTIVATION))

# "LSTM_AUTOENCODER":
model = Sequential()
model.add(Conv1D(filters=128, kernel_size=2, activation='relu', name='extractor',
                 input_shape=(Xtrain.shape[1], Xtrain.shape[2])))
model.add(Dropout(0.3))
model.add(MaxPooling1D(pool_size=2))
model.add(Bidirectional(LSTM(50, activation='relu', input_shape=(Xtrain.shape[1], Xtra
model.add(RepeatVector(10))
model.add(Bidirectional(LSTM(50, activation='relu')))
model.add(Dense(1))

# "GRU":
model = Sequential()
model.add(GRU(75, return_sequences=True, input_shape=(Xtrain.shape[1], Xtrain.shape[2]
model.add(GRU(units=30, return_sequences=True))
model.add(GRU(units=30))
model.add(Dense(units=1, activation=LAST_ACTIVATION))

# "GRU_CNN":
inp_seq = Input(shape=(Xtrain.shape[1], Xtrain.shape[2]))
x = Bidirectional(GRU(100, return_sequences=True))(inp_seq)
x = AveragePooling1D(2)(x)
x = Conv1D(100, 3, activation='relu', padding='same',
           name='extractor')(x)
x = Flatten()(x)
x = Dense(16, activation='relu')(x)
x = Dropout(0.5)(x)

out = Dense(1, activation=LAST_ACTIVATION)(x)

model = Model(inp_seq, out)
```