Supplementary material has been published as submitted. It has not been copyedited, typeset or checked for scientific content by Acta Dermato-Venereologica

APPENDIX S1 - R source code for the CNN model definition and training.

CNN, convolutional neural network.

```
target_size <- c(600, 600)
cat_list <- c("In_situ", "Invasiv")


###### DATA AUMENTATION FOR TRAINING SET ######

train_data_gen_AUG <- image_data_generator(
  rescale = 1/255,
  rotation_range = 360,
  width_shift_range = 0.2,
  height_shift_range = 0.2,
  shear_range = 0.2,
  zoom_range = 0.2,
  horizontal_flip = TRUE,
  vertical_flip = TRUE,
  fill_mode = "nearest",
)


validation_data_gen <- image_data_generator(
  rescale = 1/255
)


###### DEFINE BATCH SIZES FOR TRAINING AND VALIDATION SET ######

train_image_array_gen_AUG <- flow_images_from_directory(train_image_files_path,
                                                        train_data_gen_AUG,
                                                        target_size = target_size,
                                                        classes = cat_list,
                                                        batch_size = 10,
                                                        class_mode = "binary")

validation_image_array_gen <- flow_images_from_directory(validation_image_files_path,
                                                         validation_data_gen,
                                                         target_size = target_size,
                                                         classes = cat_list,
                                                         batch_size = 10,
                                                         class_mode = "binary")
```

```
########### MODEL DEFINITION #############
DERM_INPUT <- layer_input(shape = c(600, 600, 3), name = "DERM_INPUT")

OUTPUT_FROM_DERM_CONV <-  DERM_INPUT %>%
  layer_conv_2d(filters = 16, kernel_size = c(3, 3), activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 32, kernel_size = c(3, 3), activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 64, kernel_size = c(3, 3), activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 128, kernel_size = c(3, 3), activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 128, kernel_size = c(3, 3), activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 128, kernel_size = c(3, 3), activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2), name="LASTMAXPOOL_DERM") %>%
  layer_flatten(name="FLATTENED_FROM_DERM_CONV")
MAIN_OUTPUT <-
  OUTPUT_FROM_DERM_CONV %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 128, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid", name="MAIN_OUTPUT")
```

```
model <- keras_model(
  inputs  = c(DERM_INPUT),
  outputs = c(MAIN_OUTPUT)
)

############# COMPILE THE MODEL #############

model %>% compile(
  loss = "binary_crossentropy",
  optimizer = optimizer_rmsprop(lr = 5e-5),
  metrics = c("acc", tf$keras$metrics$AUC(name="auc"))
)


############# TRAIN THE MODEL #############

history <- model %>% fit_generator(
  train_image_array_gen_AUG,
  steps_per_epoch = 129,
  epochs = 60,
  validation_data = validation_image_array_gen,
  validation_steps = 30,
  class_weight = list("0"=1, "1"=(683/554)),
  callbacks = list(
    callback_csv_logger(filename=paste0(logspath, modelnamebest, ".txt"),
                        append=TRUE, separator="\t"),
    callback_model_checkpoint(filepath=paste0(logspath, modelnamebest),
                              save_best_only=TRUE,
                              monitor="val_auc",
                              mode="max")
  )
)
```

APPENDIX S2 - Model summary for the selected CNN model used in this investigation.

CNN, convolutional neural network.

```
_________________________________________________________________
Layer (type)                    Output Shape                Param #
=================================================================
DERM_INPUT (InputLayer)         [(None, 600, 600, 3)]        0
_________________________________________________________________
conv2d_6 (Conv2D)               (None, 598, 598, 16)         448
_________________________________________________________________
max_pooling2d_5 (MaxPooling2D)  (None, 299, 299, 16)         0
_________________________________________________________________
conv2d_7 (Conv2D)               (None, 297, 297, 32)         4640
_________________________________________________________________
max_pooling2d_6 (MaxPooling2D)  (None, 148, 148, 32)         0
_________________________________________________________________
conv2d_8 (Conv2D)               (None, 146, 146, 64)         18496
_________________________________________________________________
max_pooling2d_7 (MaxPooling2D)  (None, 73, 73, 64)           0
_________________________________________________________________
conv2d_9 (Conv2D)               (None, 71, 71, 128)          73856
_________________________________________________________________
max_pooling2d_8 (MaxPooling2D)  (None, 35, 35, 128)          0
_________________________________________________________________
conv2d_10 (Conv2D)              (None, 33, 33, 128)         147584
_________________________________________________________________
max_pooling2d_9 (MaxPooling2D)  (None, 16, 16, 128)          0
_________________________________________________________________
conv2d_11 (Conv2D)              (None, 14, 14, 128)         147584
_________________________________________________________________
LASTMAXPOOL_DERM (MaxPooling2D) (None, 7, 7, 128)            0
_________________________________________________________________
FLATTENED_FROM_DERM_CONV (Flatten) (None, 6272)              0
_________________________________________________________________
dropout_1 (Dropout)             (None, 6272)                 0
_________________________________________________________________
dense_1 (Dense)                 (None, 128)                 802944
_________________________________________________________________
MAIN_OUTPUT (Dense)             (None, 1)                    129
=================================================================
Total params: 1,195,681
Trainable params: 1,195,681
Non-trainable params: 0
_________________________________________________________________
```

APPENDIX S3

Technical information regarding program used and image processing.

The Keras library (version 2.3.1) using the Tensorflow backend (version 1.14.0) was used running on Python version 3.6.9. Model construction was performed using R version 3.5.3 (The R Foundation for Statistical Computing, Vienna, Austria) and the R-package Keras was used to call Python and its above libra- ries. XnView version 2.20 was used to scale and crop all images to quadratic shape and a resolution of 600×600 pixels (preserving the aspect ratio). All images were converted from JPEG to PNG format. A 24-bit colour depth was used (3 RGB channels with 8 bits in each channel). The computer running the training used the GPU version on the Keras/Tensorflow routines. The graphics card was a Nvidia Geforce GTX 1070 with 8 GB GPU memory using CUDA version 10.0 and cudnn version 7.6.3.30. The processor was an Intel Core i5-2400 @ 3.10 GHz with 24 GB RAM.