Algorithm 1 - Active Learning Procedure. The symbols are as follows: D_N is a data set with N samples, $H_{N\times R\times C\times 19}$ consists of heatmaps for each data sample with the dimension of $R \times C \times 19$ (46x54x19), th_{HM} is the heatmap activation threshold to be accepted as a high activation, th_{KP} is the threshold for the number of keypoints, $Resnet50$ is the model used for transfer learning, and L is the number diverse samples to be selected for the annotation.

Input: Data D_N, Feature Extraction Model $ResNet50$, Heatmaps $H_{N\times R\times C\times 19}$
Output: $FilterSetList_M$

function UncertaintySamplingModule
  Input: heatmap threshold th_{HM}, number of keypoints th_{KP}
  Initialize $FilterSetList$
  for $i = 1$ **to** N **do**
    tmp_count **= 0**
    for $j = 1$ **to** 18 **do**
      tmp_h **= ApplyNonMaxSupression(**$H_{i\times R\times C\times j}$**)**
      tmp_p **= GetHeatPeaks(**tmp_h**)**
      if $tmp\_p < th_{\mathrm{HM}}$ **then**
        $tmp\_count = tmp\_count + 1$
      end if
    end for
    if $tmp\_count > th_{\mathrm{KP}}$ **then**
      $FilterSetList.add(D_i)$
    end if
  end for
  return $FilterSetList$
end function

function FeatureExtractionModule
  Input: Data $FilterSetList_M$
  Initialize $FeatureSet_{M\times F}$
  for $i = 1$ **to** M **do**
    tmp_f **= Resnet50.fit(**$FilterSetList_i$**)**
    $FeatureSet_{i\times F}.add(tmp\_f)$
  end for
  return $FilterSetList$
end function

function DiversitySamplingModule
  Input: Data $FilterSetList_M$, Feature $FeatureSet_{M\times F}$, number of samples L
  Initialize $DissimilarMat_M$
  $DissimilarMat_M$ **= ConstructApproximateNearestNeighborTree()**
  $FilterIndices$ **=** $DissimilarMat_M$**.GetIndices(**L**)**
  return $FilterSetList_M[FilterIndices]$
end function

main Main
  UncercaintySamplingModule()
  FeatureExtractionModule()
  DiversitySamplingModule()
end main