Algorithm 1 DeepER - Identifying Matches and Non-matches

Require: A dataset $\mathcal{D}$ consisting of n pairs of tuples, $(t_1^{(i)}, t_2^{(i)})$ for $t = 1, 2, ..., n$

Ensure: Return a label vector y where $y[i] = \begin{cases} 1, \text{ if } (t_1^{(i)}, t_2^{(i)}) \text{ match} \\ 0, \text{ otherwise} \end{cases}$

for each tuple $t \in \mathcal{D}$ **do**
  for each attribute A_j of t **do**
    Pre-process and tokenize $t[A_j]$
    Convert each token $w_l \in t[A_j]$ into a word embedding $e_l \in \mathbb{R}^k$ using pre-trained GloVe embeddings
    $t[v_j]$ = composed attribute embedding obtained from either averaging or applying an LSTM on all e_l
  end for
end for
for each tuple pair $(t_1^{(i)}, t_2^{(i)}) \in \mathcal{D}$ **do**
  for each attribute j **do**
    s_j = similarity between $t_1[v_j]$ and $t_2[v_j]$. Can be scalar or vector depending on compositional method used to generate v_j
  end for
  concatenate all s_j into single distributed similarity vector s
  Use densely connected trained neural network $\mathcal{N}$ to classify s as $\mathcal{N}(s) = \begin{cases} 1, \text{ if } (t_1, t_2) \text{ match} \\ 0, \text{ otherwise} \end{cases}$
  $y[i] = \mathcal{N}(s)$
end for
return y