Learning to Align the Source Code to the Compiled Object Code – Supplementary Material

1. The Effect of Optimization

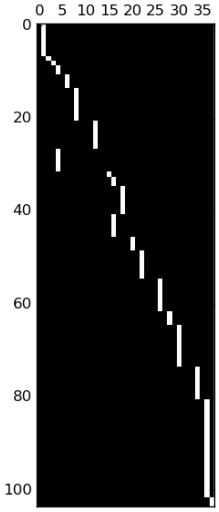
Fig. 1 demonstrates the effect of code optimization to the translated object code and the additional challenge it presents to our neural alignment network. The C code in (a) is being compiled without optimization (b), and with optimization levels 1–3 (c–e). As can be seen, optimization drastically reduces the length of the object code (N) from over a hundred statements in the unoptimized compilation to 26 statements in all three levels of optimization. The optimization also results in parts of the C code that are not covered by any statement of the object code, due to precomputation at compilation time. In general, the alignment is non-monotonous, and is less and less so as the level of optimization increases.

2. Sample Alignment Results

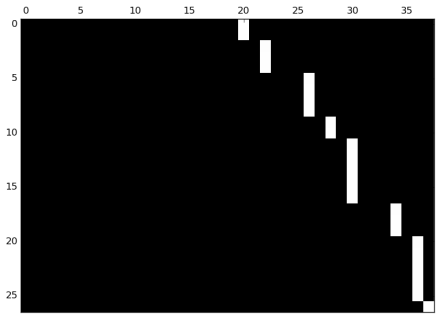
A few alignment prediction results are shown in Fig. 2, where the soft- and the hard-predictions output by our Convolutional Grid Decoder are displayed side by side with the ground truth. It is evident that the soft predictions themselves are mostly confident with values that are concentrated around 0 and 1, and that the hard-predictions closely match the ground truth alignments.

```
 1  int func1(int a1, int a2, int a3,
               int a4, int a5)
 2  {
 3   int v6, v7, v8, v9, v10, v11 = 0;
 4   v6 = 0;
 5   for (v7 = 0; v7++; v7<a5)
 6   {
 7    if (a2 < a5)
 8    {
 9     v6 += (a3 - a5) + a2;
10    }
11    else
12    {
13     v6 += a1 * (a5 - a3);
14    }
15   }
16   v8 = 0;
17   for (v9 = 0; v9++; v9<a5)
18   {
19    v8 += a1 * a5 + a3;
20   }
21   if (a4 < a1)
22   {
23    v10 = a2 - a5;
24   }
25   else
26   {
27    v10 = a2 * a3 * a5 + a3;
28   }
29   if (v6 < a3)
30   {
31    v11 = a3 * a5 * (a1 - a3);
32   }
33   else
34   {
35    v11 = ((a4 + a5) + a3) - a1;
36   }
37   return v10+v11+a1+a3+a2+a5+
            a4+v6+v7+v8+v9;
38  }
```

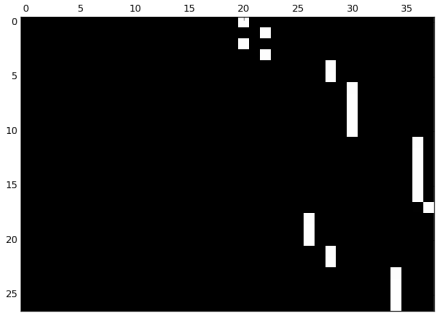
(a)



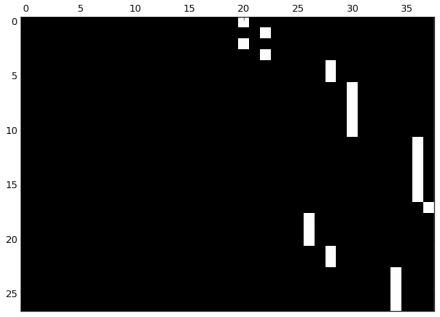
(b)



(c)

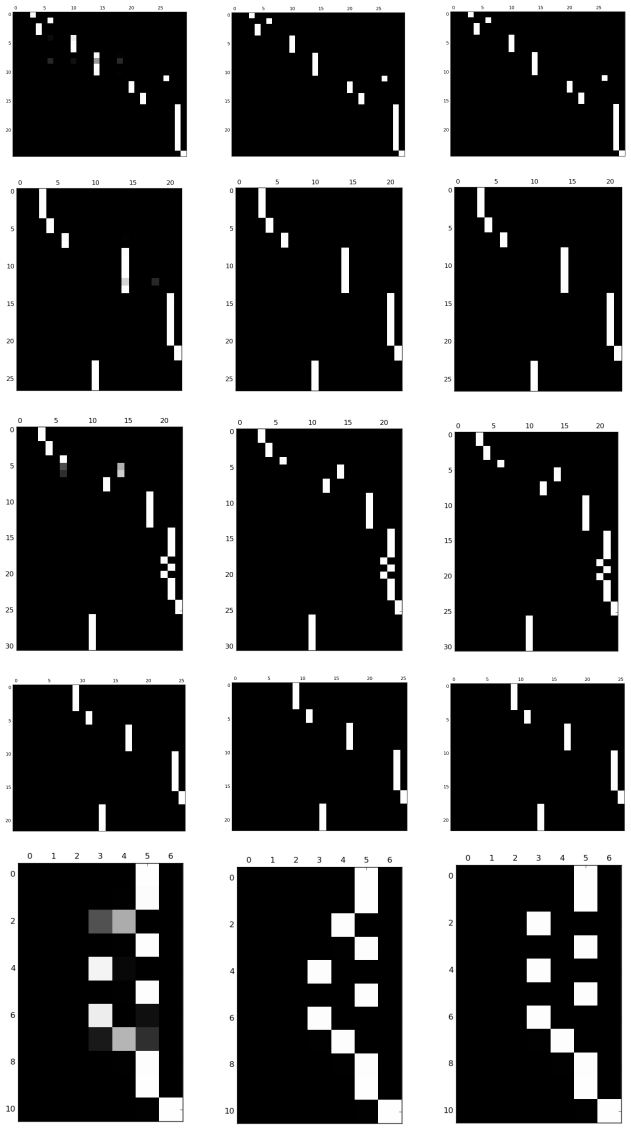


(d)



(e)

Figure 1. The effect of compiler optimization levels on the compiled object code. (a) A sample C function. (b-e) The alignment matrices for the object code that results from compiling the C code using the GCC compiler with optimization levels 0,1,2,3 respectively (the object code itself is not shown). The alignment is much less monotonous post-optimization, and the optimization results in many source code statements that have been precomputed and are not aligned. Also, for this specific code, the results of optimization levels 2 and 3 are identical.



(a) (b) (c)

Figure 2. Samples of alignments predicted by our proposed Convolutional Grid Decoder. Each row is one sample. The first sample is using `-O1` optimization; The next two samples employ `-O2`, and the rest employ `-O3`. Each matrix cell varies between 0 (black) to 1 (white). (a) The soft prediction of the alignment. (b) The predicted hard-alignment. (c) Ground truth. The soft predictions are mostly certain and the hard predictions match almost completely the ground truth.