Feature and Circuit Identification through Sparse Eigendecomposition

Some terms

Model

The output of model f (of a given architecture), depends on two inputs - a set of parameters W and a set of feature x.

After training a given model, we end up with a set of weights/parameters W_0.

Divergence

The divergence metric (not called loss, because I use that term later) should describe how much a model's output changes (for a given sample) when it's parameters are altered.

In the case of a transformer, we can compute the normalized KL-divergence. The KL-divergence describes the divergence between probability distributions and we can normalize this metric to have a minimum of zero using the following equation (note that the unnormalized KL divergence does not necessarily have a minimum at 0). We could potentially use a normalized cross entropy here as well.

$ D(f, x, W) = KL(f(W, x), f(W_0, x)) - KL(f(W_0, x), f(W_0, x)) $

In the case of a regression problem, we can use the mean squared error loss as our divergence metric. We don't need to normalize it because MSE does have a guarenteed minimum at 0.

$ D(f, x, W) = MSE(f(W, x), f(W_0, x)) $

Sample-level Hessian

We define the sample-level hessian as the second derivative of a model's loss with respect a set of parameters w, evaluated at $W0$ and X.

$ H(x,w) = \nabla^{2}_{w} D(f, x, W) $

Using the Hessian, we can compute the second derivative of the loss with respec to any direction in weight/parameter space (given by vector u) by:

$ \nabla^{2}_{u} D(f, x, W) = u(u H(x,w))^T $

Learning feature vectors

Our goal is to identify important directions (${u}$ or U) in parameter space of this Hessian where: (1) For some samples, $\nabla^{2}_{u} D$ *is very high. Moving parameters in the direction of u dramatically changes the output of the model. (2) There is low sample-level interference between each u vector. u vectors should either be nearly orthogonal, or if they* *have high cosine similarities, they should not both have high* $\nabla^{2}_{u} D$ for a given sample.

We wish to learn a set of u-vectors (U) that satisfy these conditions. We will therefore minimize two losses.

To satisfy the first goal we minimize:

$$ L_{\text{low interference}}(x) = \sum_{u}(\nabla^{2}_{u} D(f, x, W_0))^2 $$

To satisfy the second goal, we also want to minimize:

$$ L_{\text{low interference}}(x) = |\nabla^{2}_{u1}D \space u_1 (\nabla^{2}_{u2}D \space u_2)^{T} |_{u_1, u_2} $$

Optimization tricks

Optimizing losses that depend on a full hessian is computationally expensive. We use two tricks to minimize these losses.

Jacobian-vector products

First, Instead of forming the full hessian, or full jacobian, we can compute nested jacobian-vector products.

$$ \nabla^{2}_{u} D(f, X, W_0) = \nabla_{w} (\nabla_{w} D(f, X, W_0) \space u) \space u $$

Bi-level optimization

We use two separate optimization loops and perform bi-level optimization.

```
For x_batch in X:
    L1 = 0
    d2D_all = []

    # First optimization step.

    For u_batch in U:

        # Compute steep hessian loss.
        1. d2D(U) = second derivative of D(f, x, W_0) with respect to
u_batch.
        2. L1 = L1 + sum(d2D^2) # Compute first loss.
        3. d2D_all = d2D_all.append(d2D(U))


    U = U + L1*step    # Update U

    # Second optimization step.
    L2 = || (U d2D_all) (U d2D_all)^T  || # Compute low-interference loss.
    U = U + (L1 + \lambda L2)*step # Update U.
```

Note that in our first optimization loop (minimizing $L_{\text{low itnerference}}$), we can compute L in batches of U.

Toy models

XOR Model

We train a VERY simple neural network to learn the "XOR" function. The NN network consists of a single hidden layer of 2 nodes, with Gelu activation functions. The output of the hidden layer is summed to get a final output. There are 4 parameters today (2 weights an 2 biases). The training data for this network looks like:

```
[0, 1] --> 1
[1, 0] --> 1
[0, 0] --> 0
[1, 1] --> 0
```

Toy model's of superposition

We train a TMS (autoencoder 5 features, 2 hidden dimensions, Relu activation, with W_in and W_out as transposes). We set the features to uniform random numbers bewteen 0 and 1, with 5% sparsity. The TMS model successfully represents the features in pentagonal superposition.

Transformers

We use the tiny-stories-1M transformer.

Results

XOR Model

The eigenmodel successfully finds features that are most highly activated by [0,1], [1,1], and [1,0].

```
feature_idx
[sample input values] -> feature value

feature 0
[1. 1.] -> 2.3252432
[1. 0.] -> 1.0929958
[0. 1.] -> 0.61902356
[0. 0.] -> 0.273664

feature 1
[0. 1.] -> 2.670694
[1. 0.] -> 0.65577537
[1. 1.] -> 0.52892005
[0. 0.] -> 0.15629882
```

```
feature 2
[1. 0.] -> 2.021866
[1. 1.] -> 0.39354768
[0. 1.] -> 0.035297774
[0. 0.] -> 0.011965705
```

TMS

The most highly activating samples are the following:

```
feature_idx
[sample input values] -> feature value

feature 0
[0.    0.779 0.    0.    0.948] -> 3.342
[0.    0.833 0.    0.    0.35 ] -> 2.111
[0.    0.999 0.    0.    0.   ] -> 1.862
[0.    0.997 0.    0.    0.   ] -> 1.857
[0.    0.272 0.    0.    0.944] -> 1.836

feature 1
[0.888 0.    0.    0.835 0.   ] -> 3.406
[0.795 0.    0.    0.812 0.   ] -> 3.075
[0.796 0.    0.    0.78  0.   ] -> 2.977
[0.472 0.    0.    0.881 0.   ] -> 2.455
[0.285 0.    0.    0.91  0.   ] -> 2.101

feature 2
[0.758 0.    0.    0.    0.975] -> 2.078
[0.641 0.    0.    0.    0.698] -> 1.383
[0.517 0.    0.    0.    0.744] -> 1.295
[0.    0.    0.    0.    0.999] -> 1.274
[0.    0.    0.    0.    0.992] -> 1.259

feature 3
[0.758 0.    0.    0.    0.975] -> 2.516
[0.641 0.    0.    0.    0.698] -> 1.724
[0.979 0.    0.    0.    0.   ] -> 1.666
[0.971 0.    0.    0.    0.   ] -> 1.646
[0.517 0.    0.    0.    0.744] -> 1.546

feature 4
[0.126 0.    0.942 0.477 0.   ] -> 2.439
[0.    0.    0.984 0.    0.   ] -> 2.241
[0.    0.    0.971 0.    0.   ] -> 2.199
[0.   0.   0.97 0.   0.  ] -> 2.196
[0.    0.    0.967 0.    0.   ] -> 2.187
```

If we only consider completely sparse samples, the top features are the following.

```
feature_idx
[sample input values] -> feature value

feature 0
[0.    0.999 0.    0.    0.   ] -> 1.862
[0.    0.997 0.    0.    0.   ] -> 1.857
[0.    0.981 0.    0.    0.   ] -> 1.816
[0.    0.971 0.    0.    0.   ] -> 1.793
[0.    0.957 0.    0.    0.   ] -> 1.758

feature 1
[0.    0.    0.    0.998 0.   ] -> 1.728
[0.    0.    0.    0.996 0.   ] -> 1.724
[0.    0.    0.    0.995 0.   ] -> 1.721
[0.    0.    0.    0.991 0.   ] -> 1.712
[0.   0.   0.   0.97 0.  ] -> 1.662

feature 2
[0.    0.    0.    0.    0.999] -> 1.274
[0.    0.    0.    0.    0.992] -> 1.259
[0.    0.    0.    0.    0.969] -> 1.213
[0.    0.    0.    0.    0.952] -> 1.181
[0.    0.    0.    0.    0.938] -> 1.154

feature 3
[0.979 0.    0.    0.    0.   ] -> 1.666
[0.971 0.    0.    0.    0.   ] -> 1.646
[0.915 0.    0.    0.    0.   ] -> 1.523
[0.901 0.    0.    0.    0.   ] -> 1.493
[0.897 0.    0.    0.    0.   ] -> 1.485

feature 4
[0.    0.    0.984 0.    0.   ] -> 2.241
[0.    0.    0.971 0.    0.   ] -> 2.199
[0.   0.   0.97 0.   0.  ] -> 2.196
[0.    0.    0.967 0.    0.   ] -> 2.187
[0.    0.    0.958 0.    0.   ] -> 2.157
```

Transformer

Here are some results using just the weights in transformer.blocks.4.attn.W_K (4096 weights) in the tiny-stories-1M model. Here are some results from a model trained very briefly (<10 epochs, on only 500 token sets) with only 10 features pulled out. Results on data the eigenmodel was not trained on.

```
Feature description (by me)
tokens (activating token bolded) -> Feature value

Word/punctuation after a name
upon a time, there was a kind man named Tom**.** Tom had a big -> .
```

```
(Value: 42.845)
zoo.Once upon a time, there was a little boy named Timmy**.** -> . (Value:
40.725)
upon a time, there was a woman named Lily.** She** loved to go for ->  She
(Value: 38.833)
Once upon a time, there was a little boy named Timmy**.** Timmy -> .
(Value: 38.542)
Once upon a time, there was a little boy named Timmy**.** Timmy -> .
(Value: 38.542)

Once upon a time
upon a time, there was a woman named Lily.** She** loved to go for ->  She
(Value: 64.518)
could always ask the black cat.Once upon a time**,** there was a small ->
, (Value: 56.562)
loved ones. The end.Once upon a time**,** there was a little girl -> ,
(Value: 50.217)
enjoyed the sunshine.Once upon a time**,** there was a little boy -> ,
(Value: 46.414)
street and always looked out for banana peels.Once upon a time**,** there
-> , (Value: 46.287)

Tim
and trucks. One day,** Tim**my's dad took him to the park to ->  Tim
(Value: 69.034)
lost and felt sad.newlinenewlineOne day,** Tim**my's friend, a ->  Tim
(Value: 66.466)
toys. The next day,** Tim**my went to his friend's house and said ->  Tim
(Value: 62.213)
was playing just as well as before.** Tim**my was so happy with his new ->
Tim (Value: 57.687)
the noisy trunk.Once upon a time, there was a boy named** Tim**my ->  Tim
(Value: 54.684)

?
park. One day, she saw a big statue of a dog**.** It was -> . (Value:
30.722)
't like that. He wanted his truck to be the** best**. They argued for ->
best (Value: 27.736)
every day. One day, he saw a little bird on a branch**.** The -> . (Value:
26.222)
newlinenewlineOne day, Timmy saw a black cat in his backyard**.** The -> .
(Value: 22.828)
ily said, "Let's ask for help!" They saw a man** and** asked ->  and
(Value: 22.407)

Gender / polysemanic
upon a time, there was a woman named Lily.** She** loved to go for ->  She
(Value: 59.775)
Once upon a time, there was an old man.** He** liked to read magazines ->
He (Value: 39.421)
the noisy trunk.Once upon a time, there was a boy named** Tim**my ->  Tim
(Value: 38.685)
summer.Once upon a time, there was an old lady.** She** was very ->  She
```

```
(Value: 37.195)
came in to see what was going on.newlinenewlineShe told** Tim**my that ->
Tim (Value: 36.652)

saw + polysemantic
the park every day. newlinenewlineOne day, the man** saw** the woman ->
saw (Value: 39.632)
the dark hole in the ground and he fell in.newlinenewline**Tim**my tried -
> Tim (Value: 30.418)
and went inside the tent. newlinenewlineThe lion** saw** a man with a ->
saw (Value: 28.665)
Allowed." Timmy was sad because he wanted Max to come with them**.** -> .
(Value: 28.455)
's mommy lifted her up so she could see over it. She** saw** her ->  saw
(Value: 28.105)

Tim + "then on?"
toy tools. From that day on,** Tim**my learned the importance of sharing
and ->  Tim (Value: 75.689)
was playing just as well as before.** Tim**my was so happy with his new ->
Tim (Value: 63.231)
at them. He even got to touch a baby shark! After that,** Tim** ->  Tim
(Value: 59.772)
then, his mom came in and asked what was wrong.** Tim**my told her ->  Tim
(Value: 58.194)
the dark hole in the ground and he fell in.newlinenewline**Tim**my tried -
> Tim (Value: 57.360)



Animals?
named Timmy. He had a big, brown dog named Max**.** Max loved -> . (Value:
54.549)
park. One day, she saw a big statue of a dog**.** It was -> . (Value:
43.561)
She loved to walk on the trail with her dog, Max**.** Max was very -> .
(Value: 39.359)
newlinenewlineOne day, Timmy saw a black cat in his backyard**.** The -> .
(Value: 36.026)
and wanted to make him feel better. Tommy** told** Sammy a joke and Sammy
laughed ->  told (Value: 34.926)



was playing just as well as before.** Tim**my was so happy with his new ->
Tim (Value: 55.711)
part of the park where dogs were allowed.** Tim**my was happy again
because he ->  Tim (Value: 38.506)
then, his mom came in and asked what was wrong.** Tim**my told her ->  Tim
(Value: 37.558)
should answer with courage. So,** Tim**my took a deep breath and stood up
->  Tim (Value: 34.767)
came in to see what was going on.newlinenewlineShe told** Tim**my that ->
Tim (Value: 33.680)
```

```
came in to see what was going on.newlinenewlineShe told** Tim**my that ->
Tim (Value: 46.779)
at them. He even got to touch a baby shark! After that,** Tim** ->  Tim
(Value: 34.167)
the dark hole in the ground and he fell in.newlinenewline**Tim**my tried -
> Tim (Value: 30.990)
part of the park where dogs were allowed.** Tim**my was happy again
because he ->  Tim (Value: 30.866)
that they were safe.newlinenewlineBut the next day,** Tim**my didn't ->
Tim (Value: 30.029)
```