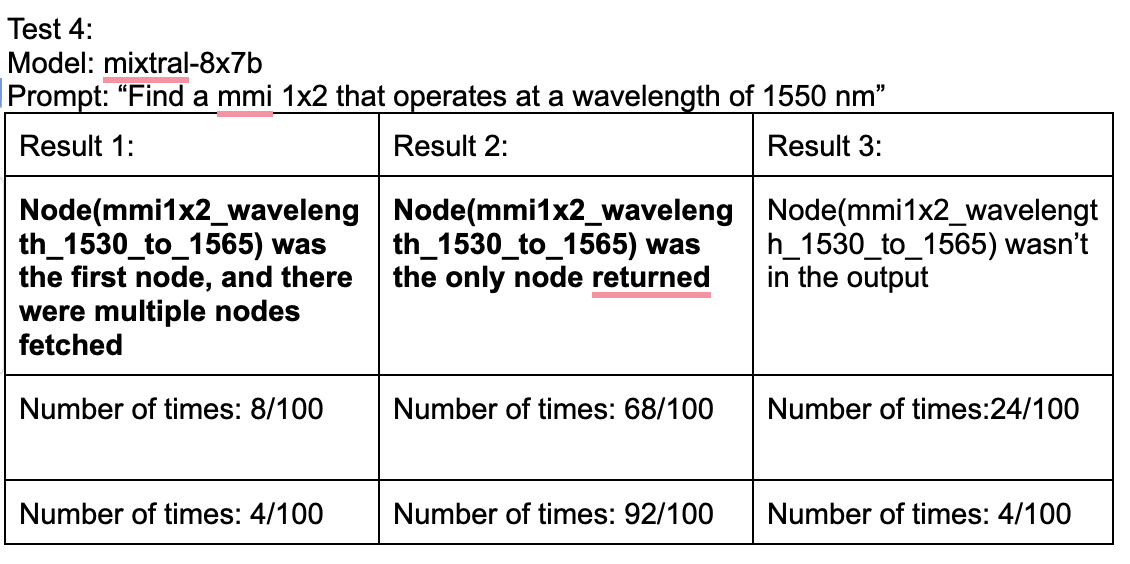
**Summary:**

each prompt was run 100 times and the results were recorded in a table, each column in the table represents one type of result, summing up the number in all the columns will obtain 100 in total.

Fibers may return multiple nodes for a given user query, so the “correct” results in this case would be if the desired node was the only node returned, or if it’s the first node among the multiple nodes returned. I also highlighted the correct cases in bold in the table

example:



If we look at the boxed row, in the 100 times, there are 8 times when the correct node is the first node among multiple nodes returned (result 1), 68 times when the only node returned is the correct node (result 2), and 24 times when the correct node wasn’t there (result 3). 8+68+24=100

similar to the row below, the total number of times is 4+92+4=100 times. The reason I ran another 100 times was mainly because I updated the search method and needed to rerun 100 tests to make sure the success rate increased

**Mistral AI model test results**

Test 1:

Model: mixtral-8x7b

Prompt: “I need a grating coupler design for TE polarized light operating at 1550 nm.”

| Result 1: | Result 2: | Result 3: |
| --- | --- | --- |
| **Node(te\_grating\_coupler\_wavelength\_1530\_to\_1565) was the first node, and there were multiple nodes fetched** | **Node(te\_grating\_coupler\_wavelength\_1530\_to\_1565) was the only node returned** | Node(te\_grating\_coupler\_wavelength\_1530\_to\_1565) was not returned |
| Number of times: 24/100 | Number of times: 76/100 | Number of times: 0/100 |
| Number of times: 0/100 | Number of times: 97/100 | Number of times: 3/100 |

Test 2:

Model: mixtral-8x7b

Prompt: “Give me the design of a TE grating coupler that couples 1310 nm light on to the chip.”

| Result 1: | Result 2: |
| --- | --- |
| **[Node(te\_grating\_coupler\_wavelength\_1260\_to\_1360)]**  **was the only node returned** | [Node(te\_grating\_coupler\_wavelength\_1260\_to\_1360)] was not the first node among multiple nodes that returned |
| Number of times: 90/100 | Number of times: 10/100 |
| Number of times: 90/100 | Number of times: 10/100 |

Test 3:

Model: mixtral-8x7b

Prompt: “Give me a design for a 1x2 MMI for 1310 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1260\_to\_1360) was the first node, and there were multiple nodes fetched** | **Node(mmi1x2\_wavelength\_1260\_to\_1360) was the only node returned** |
| Number of times: 6/100 | Number of times: 94/100 |

Test 4:

Model: mixtral-8x7b

Prompt: “Find a mmi 1x2 that operates at a wavelength of 1550 nm”

| Result 1: | Result 2: | Result 3: |
| --- | --- | --- |
| **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the first node, and there were multiple nodes fetched** | **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the only node returned** | Node(mmi1x2\_wavelength\_1530\_to\_1565) wasn’t in the output |
| Number of times: 8/100 | Number of times: 68/100 | Number of times:24/100 |
| Number of times: 4/100 | Number of times: 92/100 | Number of times: 4/100 |

Test 5:

Model: mixtral-8x7b

Prompt: “Design a splitter tree operating at 1550nm”

| Result 1: | Result 2: | Result 3: |
| --- | --- | --- |
| **Node(splitter\_tree\_wavelength\_1530\_to\_1565) was the first node, and there were multiple nodes fetched** | **Node(splitter\_tree\_wavelength\_1530\_to\_1565) was the only node returned** | Node(splitter\_tree\_wavelength\_1530\_to\_1565) was not the first node among the multiple nodes fetched |
| Number of times: 93/100 | Number of times: 6/100 | Number of times:1/100 |

Test 6:

Model: mixtral-8x7b

Prompt: “Design a splitter tree operating at 1310nm”

| Result 1: | Result 2: |
| --- | --- |
| **Node(splitter\_tree\_wavelength\_1530\_to\_1565) was the first node, and there were multiple nodes fetched** | **Node(splitter\_tree\_wavelength\_1530\_to\_1565) was the only node returned** |
| Number of times: 1/100 | Number of times: 99/100 |

Test 7:

Model: mixtral-8x7b

Prompt: “Develop a layout for an MZI at 1310 nm”

| Result 1: | Result 2: | Result 3: | Result 4: |
| --- | --- | --- | --- |
| **Node(mzi\_wavelength\_1260\_to\_1360) was the first node, and there were multiple nodes fetched** | **Node(mzi\_wavelength\_1260\_to\_1360) was the only node returned** | Node(mzi\_wavelength\_1260\_to\_1360) was not the first node among the multiple nodes fetched | Node(mzi\_wavelength\_1260\_to\_1360) was not returned |
| Number of times: 6/100 | Number of times:89/100 | Number of times:3/100 | Number of times: 2/100 |

Test 8:

Model: mixtral-8x7b

Prompt: “find a design for an MZI operating at 1550 nm”

| Result 2: | Result 4: |
| --- | --- |
| **Node(mzi\_wavelength\_1530\_to\_1565) was the only node returned** | Node(mzi\_wavelength\_1530\_to\_1565)was not returned |
| Number of times:97/100 | Number of times: 3/100 |

**LLAMA2-70B-chat test results**

Test 8:

Model: llama2-70B

Prompt: “Find a mmi 1x2 that operates at a wavelength of 1550 nm”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the only node returned** | Node(mmi1x2\_wavelength\_1530\_to\_1565) wasn’t in the output |
| Number of times: 72/100 | Number of times: 28/100 |

Test 9:

Model: llama2-70B

Prompt: “Give me the design of a TE grating coupler that couples 1310 nm light on to the chip.”

| Result 1: | Result 2: |
| --- | --- |
| [Node(te\_grating\_coupler\_wavelength\_1260\_to\_1360)] was not returned | **[Node(te\_grating\_coupler\_wavelength\_1260\_to\_1360)]**  **was the only node returned** |
| Number of times: 99/100 | Number of times: 1/100 |

Test 10:

Model: llama2-70B

Prompt: “I need a grating coupler design for TE polarized light operating at 1550 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(te\_grating\_coupler\_wavelength\_1530\_to\_1565) was the first node, and there were multiple nodes fetched** | **Node(te\_grating\_coupler\_wavelength\_1530\_to\_1565)was the only node returned** |
| Number of times: 2/100 | Number of times: 98/100 |

**Adding additional mmi node information into the tree:**

Mmi 1x2 – 1260nm to 1360nm:

| """  ## Multimode interferometer (MMI), also called an mmi,  Properties of mmi: silicon based , passive element , power splitter , dichroic\_filter  """  """  ### mmi 1x2, or 1x2 mmi,  operating wavelength ranges from [1260nm to 1360nm], this device won't work at wavelength out of this range.  Properties of this device include: silicon based , passive element , power splitter , number of ports equals 3 , dichroic\_filter  """  def mmi1x2\_wavelength\_1260\_to\_1360():  # creates a 1x2 mmi (multi-mode interferometer) that only operates at wavelength from 1260nm to 1360nm  # this device won't work for wavelength out of this range  # operating wavelength includes: 1260nm, 1261nm, 1262nm, 1263nm, 1264nm, 1265nm,  # 1266nm, 1267nm, 1268nm, 1269nm, 1270nm, 1271nm,  # 1272nm, 1273nm, 1274nm, 1275nm, 1276nm, 1277nm,  # 1278nm, 1279nm, 1280nm, 1281nm, 1282nm, 1283nm,  # 1284nm, 1285nm, 1286nm, 1287nm, 1288nm, 1289nm,  # 1290nm, 1291nm, 1292nm, 1293nm, 1294nm, 1295nm,  # 1296nm, 1297nm, 1298nm, 1299nm, 1300nm, 1301nm,  # 1302nm, 1303nm, 1304nm, 1305nm, 1306nm, 1307nm,  # 1308nm, 1309nm, 1310nm, 1311nm, 1312nm, 1313nm,  # 1314nm, 1315nm, 1316nm, 1317nm, 1318nm, 1319nm,  # 1320nm, 1321nm, 1322nm, 1323nm, 1324nm, 1325nm,  # 1326nm, 1327nm, 1328nm, 1329nm, 1330nm, 1331nm,  # 1332nm, 1333nm, 1334nm, 1335nm, 1336nm, 1337nm,  # 1338nm, 1339nm, 1340nm, 1341nm, 1342nm, 1343nm,  # 1344nm, 1345nm, 1346nm, 1347nm, 1348nm, 1349nm,  # 1350nm, 1351nm, 1352nm, 1353nm, 1354nm, 1355nm,  # 1356nm, 1357nm, 1358nm, 1359nm, 1360nm  # return gf.components.mmi1x2(width\_taper=0.5, length\_taper=10.0, length\_mmi=1.8,  # width\_mmi=1.5, gap\_mmi=0.3, width=0.41, cross\_section='xs\_sc')  name = 'mmi1x2'  properties = dict(  silicon=True,  passive=True,  power\_splitter=True,  ports=3,  narrowband=True,  dichroic\_filter=True  )  settings = dict(  width=0, # input and output straight width. Defaults to cross\_section width.  width\_taper=1.4, # interface between input straights and mmi region.  length\_taper=[2, 15], # into the mmi region.  length\_mmi=[2, 15] , # in x direction.  width\_mmi=[2, 6], # in y direction.  gap\_mmi=[0.25, 0.25] , # gap between tapered wg.  taper=0, # taper function.  cross\_section='xs\_sc'  )  wavelength = [1.26, 1.36]  polarization = ['te']  nominal\_sparam = dict( # power transmission in linear  S11=0,  S22=0,  S33=0,  S12=0.5,  S21=0.5,  S13=0.5,  S31=0.5,  S23=0,  S32=0,  )  # length\_mmi  # <------>  # \_\_\_\_\_\_\_\_  # | |  # | \\_\_  # | \_\_ o2  # \_\_/ /\_ \_ \_ \_  # o1 \_\_ | \_ \_ \_ \_| gap\_mmi  # \ \\_\_  # | \_\_ o3  # | /  # |\_\_\_\_\_\_\_\_|  # <->  # length\_taper  # import gdsfactory as gf  # c = gf.components.mmi1x2(width\_taper=1.0, length\_taper=10.0, length\_mmi=5.5, width\_mmi=2.5, gap\_mmi=0.25, cross\_section='xs\_sc')  # c.plot() |
| --- |

Test 1:

Model: mixtral-8x7b

Prompt: “Give me a design for a 1x2 MMI for 1310 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1260\_to\_1360) was the first node, and there were multiple nodes fetched** | **Node(mmi1x2\_wavelength\_1260\_to\_1360) was the only node returned** |
| Number of times: 2/100 | Number of times: 98/100 |

Test 2:

Model: mixtral-8x7b

Prompt: “Find a mmi 1x2 that operates at a wavelength of 1550 nm”

| Result 1: | Result 2: |
| --- | --- |
| Node(mmi1x2\_wavelength\_1530\_to\_1565) was not returned | **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the only node returned** |
| Number of times: 3/100 | Number of times: 97/100 |

**LLAMA3-70B-instruct test results**

Test 1:

Model: llama3-70B- instruct

Prompt: “I need a grating coupler design for TE polarized light operating at 1550 nm.”

| Result: |
| --- |
| **Node(te\_grating\_coupler\_wavelength\_1530\_to\_1565) was the only node returned** |
| Number of times:100/100 |

Test 2:

Model: llama3-70B- instruct

Prompt: “Give me the design of a TE grating coupler that couples 1310 nm light on to the chip”

| Result 1: | Result 2: |
| --- | --- |
| **[Node(te\_grating\_coupler\_wavelength\_1260\_to\_1360)]**  **was the only node returned** | [Node(te\_grating\_coupler\_wavelength\_1260\_to\_1360)] was not the first node among multiple nodes returned |
| Number of times: 100/100 | Number of times: 0/100 |

Test 3:

Model: llama3-70B- instruct

Prompt: “Give me a design for a 1x2 MMI for 1310 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1260\_to\_1360) was the first node, and there were multiple nodes fetched** | **Node(mmi1x2\_wavelength\_1260\_to\_1360) was the only node returned** |
| Number of times: 0/100 | Number of times: 100/100 |

Test 3:

Model: llama3-70B- instruct

Prompt: “Find a mmi 1x2 that operates at a wavelength of 1550 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the only node returned** | Node(mmi1x2\_wavelength\_1530\_to\_1565) wasn’t in the output |
| Number of times: 0/50 | Number of times: 50/50 |

Test 4:

Model: llama3-70B- instruct

Prompt: “Find a 1x2 mmi that operates at a wavelength of 1550 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the only node returned** | Node(mmi1x2\_wavelength\_1530\_to\_1565) wasn’t in the output |
| Number of times: 50/50 | Number of times: 0/50 |

Note here the only difference in the 2 queries was 1x2 mmi vs. mmi 1x2, but the results were different.

Update: this problem was fixed by adding more information to the mmi parent node in photonic\_component.py

Before:



After:



Redo test 3:

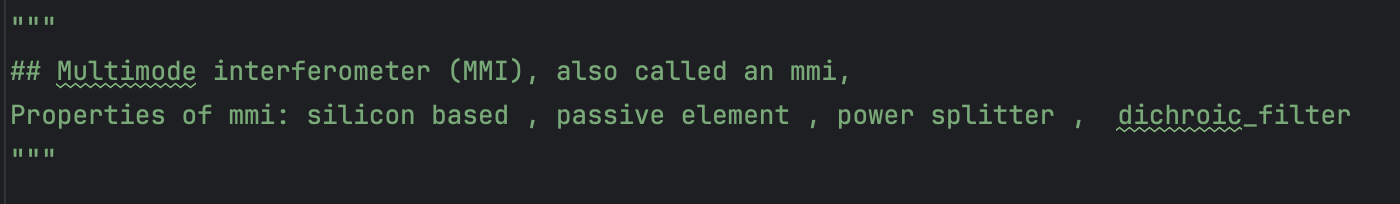
Model: llama3-70B- instruct

Prompt: “Find a mmi 1x2 that operates at a wavelength of 1550 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the only node returned** | Node(mmi1x2\_wavelength\_1530\_to\_1565) wasn’t in the output |
| Number of times: 98/100 | Number of times: 2/100 |

**Adding node labels to MMI**

adding node labels to MMI (silicon based , passive element , power splitter , dichroic\_filter)



we added some properties of mmi and wanted to see whether the LLM can return the desired node when we ask it a more generic question, such as to return a “power splitter” instead of “mmi 1x2” directly

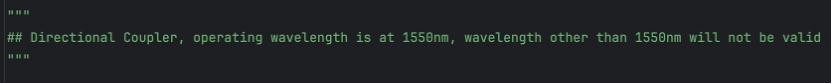
Test 1:

Model: llama3-70B- instruct

Prompt: “Find a power splitter that operates at a wavelength of 1550 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the first node among the multiple nodes** | Node(mmi1x2\_wavelength\_1530\_to\_1565) wasn’t in the output.  Note: in all of the results where the desired node is not in the output, Node(directional\_coupler\_wavelength\_1550) was the first node |
| Number of times: 83/100 | Number of times: 17/100 |
| Number of times: 100/100 | Number of times: 0/100 |

This is the node label for the directional coupler, note that we didn’t specify it to be a powers splitter as we did with MMI



Test 2:

Model: mixtral-8x7b

Prompt: “Find a power splitter that operates at a wavelength of 1550 nm.”

| Result 1: | Result 2: |
| --- | --- |
| **Node(mmi1x2\_wavelength\_1530\_to\_1565) was the first node among the multiple nodes** | Node(mmi1x2\_wavelength\_1530\_to\_1565) wasn’t in the output.    Note: in all of the result where the desired node is not in the output, Node(directional\_coupler\_wavelength\_1550) was the first node |
| Number of times: 100/100 | Number of times: 0/100 |
| Number of times: 0/100 | Number of times: 100/100 |

notice that for mixtral-8x7b, the result wasn’t stable enough