

# A tour of data, eval, and scaling language models with Olmo

David Heineman



slides: [davidheineman.com/slides/2026-03-olmo-methods-gatech.pdf](https://davidheineman.com/slides/2026-03-olmo-methods-gatech.pdf)

At Ai2, we build  
fully open LM recipes

 OLMo

To facilitate research and accelerate the science of LMs, we need language models that are fully open.



Transparent



Reproducible



Accessible

# OLMo is being used to...

## Attribute model behavior to pretraining data

## Inform policy by studying LM development ecosystem

## Understand the effects of fine-tuning a pretrained model

### LESS: Selecting Influential Data for Targeted Instruction Tuning

Mingzhou Xia<sup>1</sup>, Sathika Malladi<sup>1</sup>, Sachin Gururanga<sup>1</sup>, Sameer Arora<sup>1</sup>, Dhanj Chou<sup>1</sup>

#### Abstract

Instruction tuning has unlocked powerful capabilities in large language models (LLMs) using combinatorial datasets to develop general purpose chatbots. However, real-world applications often require a specialized suite of skills (e.g., reasoning). The challenge lies in identifying the most relevant data from these extensive datasets to effectively develop specific capabilities, a setting we frame as targeted instruction tuning. We propose LESS, an optimizer-aware and practically efficient algorithm to estimate data influence and perform low-rank gradient Similarity Search for instruction data selection. Crucially, LESS adapts existing influence formulations to work with the Adam optimizer and variable-length instruction data. LESS first constructs a highly readable and transferable *profiler* dastuator with low-dimensional gradient features and then selects examples based on their similarity to few-shot examples embodying a specific capability. Experiments show that training on a LESS-selected 5% of the data can outperform full-dataset training on the full dataset across diverse downstream tasks. Furthermore, the selected data is highly transferable: smaller models can be leveraged to select useful data for larger models and models from different families. Our qualitative analysis shows that our method goes beyond surface form cues to identify data that encodes the necessary reasoning skills for the intended downstream application. To facilitate future work, we release code and data to the public at <https://github.com/least-llm/less>.

<sup>1</sup>Total contribution: Princeton Language and Intelligence (PLI), Princeton University, the University of Wisconsin, University of Washington, USC, Commonwealth of Massachusetts, Xa, Sathika Malladi, coauthors, variable-length instruction data that demand the influence estimation, and (3) the large number of transferable pretraining LLMs make the comparison and transfer of gradient information extremely more interesting.

Proceeding of the 17<sup>th</sup> Annual Conference on Artificial Learning, Vienna, Austria, June 23-24, 2024. Copyright 2024 by the authors.

#### 1. Introduction

Instruction tuning has made large language models (LLMs) adapt at following human instructions (Ouyang et al., 2022) as versatile helpers (OpenAI, 2022, 2023; Anthropic, 2023; Google, 2023). Recent efforts curating highly diverse and wide-ranging instruction tuning datasets (Touvron et al., 2023; Wang et al., 2023b; Xu et al., 2023; Yu et al., 2023) have yielded remarkably strong generalization even from a small number of examples (Zhou et al., 2023). Regardless, it remains an open problem to understand how to best utilize these vast datasets.

Many real-world applications call for cultivating a specific suite of capabilities in LLMs (e.g., reasoning, skills). However, training LLMs with mixed instruction tuning datasets can hinder the development of these specific capabilities.

For example, Wang et al. (2023b) demonstrates that LLMs trained on a mix of instruction tuning datasets exhibit worse performance than those trained on a subset of the data. Additionally, considering the broad spectrum of user queries and the multitude of skills required to respond to them, there may not always be enough in-domain data available. Therefore, we hope to be able to effectively use the general instruction tuning data to improve specific capabilities. We frame this setting as *targeted instruction tuning*.

Given just a handful of examples embodying a specific capability, how can we effectively select relevant few-shot training data from a large collection of instruction datasets? We address this problem by prioritizing training on data that directly minimizes loss on a target task instead of relying on surface form features (Gururanga et al., 2020; Xu et al., 2023b). Inspired by past work estimating the influence of individual training datapoints with gradient information (Pruthi et al., 2020; Han et al., 2023), we design an optimizer-aware approach to select such data. However, straightforward application of this influence formulation faces several challenges unique to instruction tuning settings: (1) LLMs are traditionally fine-tuned with the Adam optimizer (Kingma & Ba, 2015) instead of the canonical SGD optimizer; (2) our sparse-level gradient, variable-length instruction data that demand the influence estimation; and (3) the large number of transferable pretraining LLMs make the comparison and transfer of gradient information extremely more interesting.

### Consent in Crisis: The Rapid Decline of the AI Data Commons

Shayne Longpre<sup>1</sup>, Robert Mahur<sup>1</sup>, Arif Le<sup>1</sup>, Campbell Lane<sup>1</sup>, Hamidsh Odehriwala<sup>1</sup>, William Brarwara<sup>1</sup>, Naman Dhotw<sup>1</sup>, Naman Dhotw<sup>1</sup>, Talha Shafiq<sup>1</sup>, Cole Hutter<sup>1</sup>, Kevin Klyman<sup>1</sup>, Christopher Klamm<sup>1</sup>, Hattie Schofield<sup>1</sup>, Nikhil Singh<sup>1</sup>, Manoj Cherya<sup>1</sup>, Ahmad Mhanna<sup>1</sup>, Adil Hameed<sup>1</sup>, Candace Chikara<sup>1</sup>, Du Han<sup>1</sup>, Shashank Saha<sup>1</sup>, Devika Mittal<sup>1</sup>, Dhanu Nair<sup>1</sup>, Ehsan Alghamdi<sup>1</sup>, Ericor Shipko<sup>1</sup>, Jangmo Jung<sup>1</sup>, Joanna Materzynska<sup>1</sup>, Kim Qiu<sup>1</sup>, Kashy Tewari<sup>1</sup>, Lester Mackey<sup>1</sup>, Manan Dhotw<sup>1</sup>, Minsik Jung<sup>1</sup>, Muhammad Hamada<sup>1</sup>, Niklas Muennighoff<sup>1</sup>, Sanyam Prasad<sup>1</sup>, Sanyam Prasad<sup>1</sup>, Shreya Mohapatra<sup>1</sup>, Vipul Gupta<sup>1</sup>, Vivek Shrivastava<sup>1</sup>, Yu Minh Chen<sup>1</sup>, Yuhui Li<sup>1</sup>, Gintale Young<sup>1</sup>, Luk Vilij<sup>1</sup>, Stella Biderman<sup>1</sup>, Hamish Li<sup>1</sup>, Daphne Ipelloni<sup>1</sup>, Sara Hooker<sup>1</sup>, Jack Karbura<sup>1</sup>, and Sandy Pentland<sup>1</sup>

<sup>1</sup>Team Lead, <sup>2</sup>Top Contributor, <sup>3</sup>Contributors (alphabetically), <sup>4</sup>Advisors

#### Abstract

General purpose artificial intelligence (AI) systems are built on massive swaths of public web data, assembled into corpora such as Common Crawl, and indexed. To our knowledge, we conduct the first, large-scale, longitudinal audit of the content provided for the web domains underpinning AI training corpora. Our audit of 16,000 web domains provides an expansive view of crawlable web data and how collected data use performance an absolute one-time. We observe a proliferation of AI-specific data well as good Terms of Service (ToS) web-proxies for AI. Our findings suggest that AI-specific data well as good Terms of Service (ToS) web-proxies for AI. Our findings suggest that AI-specific data well as good Terms of Service (ToS) web-proxies for AI.

### Weak-to-Strong Extrapolation Expedites Alignment

Chujie Zhang<sup>1</sup>, Zhi Wang<sup>1</sup>, Hong Ji<sup>1</sup>, Miao Huang<sup>1</sup>, Nanyang Peng<sup>1</sup>  
<sup>1</sup>The Ohio State University, <sup>2</sup>UC Berkeley, <sup>3</sup>Microsoft, <sup>4</sup>Google, <sup>5</sup>Meta, <sup>6</sup>OpenAI, <sup>7</sup>Google, <sup>8</sup>Meta, <sup>9</sup>OpenAI, <sup>10</sup>Google, <sup>11</sup>Meta, <sup>12</sup>OpenAI, <sup>13</sup>Google, <sup>14</sup>Meta, <sup>15</sup>OpenAI, <sup>16</sup>Google, <sup>17</sup>Meta, <sup>18</sup>OpenAI, <sup>19</sup>Google, <sup>20</sup>Meta, <sup>21</sup>OpenAI, <sup>22</sup>Google, <sup>23</sup>Meta, <sup>24</sup>OpenAI, <sup>25</sup>Google, <sup>26</sup>Meta, <sup>27</sup>OpenAI, <sup>28</sup>Google, <sup>29</sup>Meta, <sup>30</sup>OpenAI, <sup>31</sup>Google, <sup>32</sup>Meta, <sup>33</sup>OpenAI, <sup>34</sup>Google, <sup>35</sup>Meta, <sup>36</sup>OpenAI, <sup>37</sup>Google, <sup>38</sup>Meta, <sup>39</sup>OpenAI, <sup>40</sup>Google, <sup>41</sup>Meta, <sup>42</sup>OpenAI, <sup>43</sup>Google, <sup>44</sup>Meta, <sup>45</sup>OpenAI, <sup>46</sup>Google, <sup>47</sup>Meta, <sup>48</sup>OpenAI, <sup>49</sup>Google, <sup>50</sup>Meta, <sup>51</sup>OpenAI, <sup>52</sup>Google, <sup>53</sup>Meta, <sup>54</sup>OpenAI, <sup>55</sup>Google, <sup>56</sup>Meta, <sup>57</sup>OpenAI, <sup>58</sup>Google, <sup>59</sup>Meta, <sup>60</sup>OpenAI, <sup>61</sup>Google, <sup>62</sup>Meta, <sup>63</sup>OpenAI, <sup>64</sup>Google, <sup>65</sup>Meta, <sup>66</sup>OpenAI, <sup>67</sup>Google, <sup>68</sup>Meta, <sup>69</sup>OpenAI, <sup>70</sup>Google, <sup>71</sup>Meta, <sup>72</sup>OpenAI, <sup>73</sup>Google, <sup>74</sup>Meta, <sup>75</sup>OpenAI, <sup>76</sup>Google, <sup>77</sup>Meta, <sup>78</sup>OpenAI, <sup>79</sup>Google, <sup>80</sup>Meta, <sup>81</sup>OpenAI, <sup>82</sup>Google, <sup>83</sup>Meta, <sup>84</sup>OpenAI, <sup>85</sup>Google, <sup>86</sup>Meta, <sup>87</sup>OpenAI, <sup>88</sup>Google, <sup>89</sup>Meta, <sup>90</sup>OpenAI, <sup>91</sup>Google, <sup>92</sup>Meta, <sup>93</sup>OpenAI, <sup>94</sup>Google, <sup>95</sup>Meta, <sup>96</sup>OpenAI, <sup>97</sup>Google, <sup>98</sup>Meta, <sup>99</sup>OpenAI, <sup>100</sup>Google, <sup>101</sup>Meta, <sup>102</sup>OpenAI, <sup>103</sup>Google, <sup>104</sup>Meta, <sup>105</sup>OpenAI, <sup>106</sup>Google, <sup>107</sup>Meta, <sup>108</sup>OpenAI, <sup>109</sup>Google, <sup>110</sup>Meta, <sup>111</sup>OpenAI, <sup>112</sup>Google, <sup>113</sup>Meta, <sup>114</sup>OpenAI, <sup>115</sup>Google, <sup>116</sup>Meta, <sup>117</sup>OpenAI, <sup>118</sup>Google, <sup>119</sup>Meta, <sup>120</sup>OpenAI, <sup>121</sup>Google, <sup>122</sup>Meta, <sup>123</sup>OpenAI, <sup>124</sup>Google, <sup>125</sup>Meta, <sup>126</sup>OpenAI, <sup>127</sup>Google, <sup>128</sup>Meta, <sup>129</sup>OpenAI, <sup>130</sup>Google, <sup>131</sup>Meta, <sup>132</sup>OpenAI, <sup>133</sup>Google, <sup>134</sup>Meta, <sup>135</sup>OpenAI, <sup>136</sup>Google, <sup>137</sup>Meta, <sup>138</sup>OpenAI, <sup>139</sup>Google, <sup>140</sup>Meta, <sup>141</sup>OpenAI, <sup>142</sup>Google, <sup>143</sup>Meta, <sup>144</sup>OpenAI, <sup>145</sup>Google, <sup>146</sup>Meta, <sup>147</sup>OpenAI, <sup>148</sup>Google, <sup>149</sup>Meta, <sup>150</sup>OpenAI, <sup>151</sup>Google, <sup>152</sup>Meta, <sup>153</sup>OpenAI, <sup>154</sup>Google, <sup>155</sup>Meta, <sup>156</sup>OpenAI, <sup>157</sup>Google, <sup>158</sup>Meta, <sup>159</sup>OpenAI, <sup>160</sup>Google, <sup>161</sup>Meta, <sup>162</sup>OpenAI, <sup>163</sup>Google, <sup>164</sup>Meta, <sup>165</sup>OpenAI, <sup>166</sup>Google, <sup>167</sup>Meta, <sup>168</sup>OpenAI, <sup>169</sup>Google, <sup>170</sup>Meta, <sup>171</sup>OpenAI, <sup>172</sup>Google, <sup>173</sup>Meta, <sup>174</sup>OpenAI, <sup>175</sup>Google, <sup>176</sup>Meta, <sup>177</sup>OpenAI, <sup>178</sup>Google, <sup>179</sup>Meta, <sup>180</sup>OpenAI, <sup>181</sup>Google, <sup>182</sup>Meta, <sup>183</sup>OpenAI, <sup>184</sup>Google, <sup>185</sup>Meta, <sup>186</sup>OpenAI, <sup>187</sup>Google, <sup>188</sup>Meta, <sup>189</sup>OpenAI, <sup>190</sup>Google, <sup>191</sup>Meta, <sup>192</sup>OpenAI, <sup>193</sup>Google, <sup>194</sup>Meta, <sup>195</sup>OpenAI, <sup>196</sup>Google, <sup>197</sup>Meta, <sup>198</sup>OpenAI, <sup>199</sup>Google, <sup>200</sup>Meta, <sup>201</sup>OpenAI, <sup>202</sup>Google, <sup>203</sup>Meta, <sup>204</sup>OpenAI, <sup>205</sup>Google, <sup>206</sup>Meta, <sup>207</sup>OpenAI, <sup>208</sup>Google, <sup>209</sup>Meta, <sup>210</sup>OpenAI, <sup>211</sup>Google, <sup>212</sup>Meta, <sup>213</sup>OpenAI, <sup>214</sup>Google, <sup>215</sup>Meta, <sup>216</sup>OpenAI, <sup>217</sup>Google, <sup>218</sup>Meta, <sup>219</sup>OpenAI, <sup>220</sup>Google, <sup>221</sup>Meta, <sup>222</sup>OpenAI, <sup>223</sup>Google, <sup>224</sup>Meta, <sup>225</sup>OpenAI, <sup>226</sup>Google, <sup>227</sup>Meta, <sup>228</sup>OpenAI, <sup>229</sup>Google, <sup>230</sup>Meta, <sup>231</sup>OpenAI, <sup>232</sup>Google, <sup>233</sup>Meta, <sup>234</sup>OpenAI, <sup>235</sup>Google, <sup>236</sup>Meta, <sup>237</sup>OpenAI, <sup>238</sup>Google, <sup>239</sup>Meta, <sup>240</sup>OpenAI, <sup>241</sup>Google, <sup>242</sup>Meta, <sup>243</sup>OpenAI, <sup>244</sup>Google, <sup>245</sup>Meta, <sup>246</sup>OpenAI, <sup>247</sup>Google, <sup>248</sup>Meta, <sup>249</sup>OpenAI, <sup>250</sup>Google, <sup>251</sup>Meta, <sup>252</sup>OpenAI, <sup>253</sup>Google, <sup>254</sup>Meta, <sup>255</sup>OpenAI, <sup>256</sup>Google, <sup>257</sup>Meta, <sup>258</sup>OpenAI, <sup>259</sup>Google, <sup>260</sup>Meta, <sup>261</sup>OpenAI, <sup>262</sup>Google, <sup>263</sup>Meta, <sup>264</sup>OpenAI, <sup>265</sup>Google, <sup>266</sup>Meta, <sup>267</sup>OpenAI, <sup>268</sup>Google, <sup>269</sup>Meta, <sup>270</sup>OpenAI, <sup>271</sup>Google, <sup>272</sup>Meta, <sup>273</sup>OpenAI, <sup>274</sup>Google, <sup>275</sup>Meta, <sup>276</sup>OpenAI, <sup>277</sup>Google, <sup>278</sup>Meta, <sup>279</sup>OpenAI, <sup>280</sup>Google, <sup>281</sup>Meta, <sup>282</sup>OpenAI, <sup>283</sup>Google, <sup>284</sup>Meta, <sup>285</sup>OpenAI, <sup>286</sup>Google, <sup>287</sup>Meta, <sup>288</sup>OpenAI, <sup>289</sup>Google, <sup>290</sup>Meta, <sup>291</sup>OpenAI, <sup>292</sup>Google, <sup>293</sup>Meta, <sup>294</sup>OpenAI, <sup>295</sup>Google, <sup>296</sup>Meta, <sup>297</sup>OpenAI, <sup>298</sup>Google, <sup>299</sup>Meta, <sup>300</sup>OpenAI, <sup>301</sup>Google, <sup>302</sup>Meta, <sup>303</sup>OpenAI, <sup>304</sup>Google, <sup>305</sup>Meta, <sup>306</sup>OpenAI, <sup>307</sup>Google, <sup>308</sup>Meta, <sup>309</sup>OpenAI, <sup>310</sup>Google, <sup>311</sup>Meta, <sup>312</sup>OpenAI, <sup>313</sup>Google, <sup>314</sup>Meta, <sup>315</sup>OpenAI, <sup>316</sup>Google, <sup>317</sup>Meta, <sup>318</sup>OpenAI, <sup>319</sup>Google, <sup>320</sup>Meta, <sup>321</sup>OpenAI, <sup>322</sup>Google, <sup>323</sup>Meta, <sup>324</sup>OpenAI, <sup>325</sup>Google, <sup>326</sup>Meta, <sup>327</sup>OpenAI, <sup>328</sup>Google, <sup>329</sup>Meta, <sup>330</sup>OpenAI, <sup>331</sup>Google, <sup>332</sup>Meta, <sup>333</sup>OpenAI, <sup>334</sup>Google, <sup>335</sup>Meta, <sup>336</sup>OpenAI, <sup>337</sup>Google, <sup>338</sup>Meta, <sup>339</sup>OpenAI, <sup>340</sup>Google, <sup>341</sup>Meta, <sup>342</sup>OpenAI, <sup>343</sup>Google, <sup>344</sup>Meta, <sup>345</sup>OpenAI, <sup>346</sup>Google, <sup>347</sup>Meta, <sup>348</sup>OpenAI, <sup>349</sup>Google, <sup>350</sup>Meta, <sup>351</sup>OpenAI, <sup>352</sup>Google, <sup>353</sup>Meta, <sup>354</sup>OpenAI, <sup>355</sup>Google, <sup>356</sup>Meta, <sup>357</sup>OpenAI, <sup>358</sup>Google, <sup>359</sup>Meta, <sup>360</sup>OpenAI, <sup>361</sup>Google, <sup>362</sup>Meta, <sup>363</sup>OpenAI, <sup>364</sup>Google, <sup>365</sup>Meta, <sup>366</sup>OpenAI, <sup>367</sup>Google, <sup>368</sup>Meta, <sup>369</sup>OpenAI, <sup>370</sup>Google, <sup>371</sup>Meta, <sup>372</sup>OpenAI, <sup>373</sup>Google, <sup>374</sup>Meta, <sup>375</sup>OpenAI, <sup>376</sup>Google, <sup>377</sup>Meta, <sup>378</sup>OpenAI, <sup>379</sup>Google, <sup>380</sup>Meta, <sup>381</sup>OpenAI, <sup>382</sup>Google, <sup>383</sup>Meta, <sup>384</sup>OpenAI, <sup>385</sup>Google, <sup>386</sup>Meta, <sup>387</sup>OpenAI, <sup>388</sup>Google, <sup>389</sup>Meta, <sup>390</sup>OpenAI, <sup>391</sup>Google, <sup>392</sup>Meta, <sup>393</sup>OpenAI, <sup>394</sup>Google, <sup>395</sup>Meta, <sup>396</sup>OpenAI, <sup>397</sup>Google, <sup>398</sup>Meta, <sup>399</sup>OpenAI, <sup>400</sup>Google, <sup>401</sup>Meta, <sup>402</sup>OpenAI, <sup>403</sup>Google, <sup>404</sup>Meta, <sup>405</sup>OpenAI, <sup>406</sup>Google, <sup>407</sup>Meta, <sup>408</sup>OpenAI, <sup>409</sup>Google, <sup>410</sup>Meta, <sup>411</sup>OpenAI, <sup>412</sup>Google, <sup>413</sup>Meta, <sup>414</sup>OpenAI, <sup>415</sup>Google, <sup>416</sup>Meta, <sup>417</sup>OpenAI, <sup>418</sup>Google, <sup>419</sup>Meta, <sup>420</sup>OpenAI, <sup>421</sup>Google, <sup>422</sup>Meta, <sup>423</sup>OpenAI, <sup>424</sup>Google, <sup>425</sup>Meta, <sup>426</sup>OpenAI, <sup>427</sup>Google, <sup>428</sup>Meta, <sup>429</sup>OpenAI, <sup>430</sup>Google, <sup>431</sup>Meta, <sup>432</sup>OpenAI, <sup>433</sup>Google, <sup>434</sup>Meta, <sup>435</sup>OpenAI, <sup>436</sup>Google, <sup>437</sup>Meta, <sup>438</sup>OpenAI, <sup>439</sup>Google, <sup>440</sup>Meta, <sup>441</sup>OpenAI, <sup>442</sup>Google, <sup>443</sup>Meta, <sup>444</sup>OpenAI, <sup>445</sup>Google, <sup>446</sup>Meta, <sup>447</sup>OpenAI, <sup>448</sup>Google, <sup>449</sup>Meta, <sup>450</sup>OpenAI, <sup>451</sup>Google, <sup>452</sup>Meta, <sup>453</sup>OpenAI, <sup>454</sup>Google, <sup>455</sup>Meta, <sup>456</sup>OpenAI, <sup>457</sup>Google, <sup>458</sup>Meta, <sup>459</sup>OpenAI, <sup>460</sup>Google, <sup>461</sup>Meta, <sup>462</sup>OpenAI, <sup>463</sup>Google, <sup>464</sup>Meta, <sup>465</sup>OpenAI, <sup>466</sup>Google, <sup>467</sup>Meta, <sup>468</sup>OpenAI, <sup>469</sup>Google, <sup>470</sup>Meta, <sup>471</sup>OpenAI, <sup>472</sup>Google, <sup>473</sup>Meta, <sup>474</sup>OpenAI, <sup>475</sup>Google, <sup>476</sup>Meta, <sup>477</sup>OpenAI, <sup>478</sup>Google, <sup>479</sup>Meta, <sup>480</sup>OpenAI, <sup>481</sup>Google, <sup>482</sup>Meta, <sup>483</sup>OpenAI, <sup>484</sup>Google, <sup>485</sup>Meta, <sup>486</sup>OpenAI, <sup>487</sup>Google, <sup>488</sup>Meta, <sup>489</sup>OpenAI, <sup>490</sup>Google, <sup>491</sup>Meta, <sup>492</sup>OpenAI, <sup>493</sup>Google, <sup>494</sup>Meta, <sup>495</sup>OpenAI, <sup>496</sup>Google, <sup>497</sup>Meta, <sup>498</sup>OpenAI, <sup>499</sup>Google, <sup>500</sup>Meta, <sup>501</sup>OpenAI, <sup>502</sup>Google, <sup>503</sup>Meta, <sup>504</sup>OpenAI, <sup>505</sup>Google, <sup>506</sup>Meta, <sup>507</sup>OpenAI, <sup>508</sup>Google, <sup>509</sup>Meta, <sup>510</sup>OpenAI, <sup>511</sup>Google, <sup>512</sup>Meta, <sup>513</sup>OpenAI, <sup>514</sup>Google, <sup>515</sup>Meta, <sup>516</sup>OpenAI, <sup>517</sup>Google, <sup>518</sup>Meta, <sup>519</sup>OpenAI, <sup>520</sup>Google, <sup>521</sup>Meta, <sup>522</sup>OpenAI, <sup>523</sup>Google, <sup>524</sup>Meta, <sup>525</sup>OpenAI, <sup>526</sup>Google, <sup>527</sup>Meta, <sup>528</sup>OpenAI, <sup>529</sup>Google, <sup>530</sup>Meta, <sup>531</sup>OpenAI, <sup>532</sup>Google, <sup>533</sup>Meta, <sup>534</sup>OpenAI, <sup>535</sup>Google, <sup>536</sup>Meta, <sup>537</sup>OpenAI, <sup>538</sup>Google, <sup>539</sup>Meta, <sup>540</sup>OpenAI, <sup>541</sup>Google, <sup>542</sup>Meta, <sup>543</sup>OpenAI, <sup>544</sup>Google, <sup>545</sup>Meta, <sup>546</sup>OpenAI, <sup>547</sup>Google, <sup>548</sup>Meta, <sup>549</sup>OpenAI, <sup>550</sup>Google, <sup>551</sup>Meta, <sup>552</sup>OpenAI, <sup>553</sup>Google, <sup>554</sup>Meta, <sup>555</sup>OpenAI, <sup>556</sup>Google, <sup>557</sup>Meta, <sup>558</sup>OpenAI, <sup>559</sup>Google, <sup>560</sup>Meta, <sup>561</sup>OpenAI, <sup>562</sup>Google, <sup>563</sup>Meta, <sup>564</sup>OpenAI, <sup>565</sup>Google, <sup>566</sup>Meta, <sup>567</sup>OpenAI, <sup>568</sup>Google, <sup>569</sup>Meta, <sup>570</sup>OpenAI, <sup>571</sup>Google, <sup>572</sup>Meta, <sup>573</sup>OpenAI, <sup>574</sup>Google, <sup>575</sup>Meta, <sup>576</sup>OpenAI, <sup>577</sup>Google, <sup>578</sup>Meta, <sup>579</sup>OpenAI, <sup>580</sup>Google, <sup>581</sup>Meta, <sup>582</sup>OpenAI, <sup>583</sup>Google, <sup>584</sup>Meta, <sup>585</sup>OpenAI, <sup>586</sup>Google, <sup>587</sup>Meta, <sup>588</sup>OpenAI, <sup>589</sup>Google, <sup>590</sup>Meta, <sup>591</sup>OpenAI, <sup>592</sup>Google, <sup>593</sup>Meta, <sup>594</sup>OpenAI, <sup>595</sup>Google, <sup>596</sup>Meta, <sup>597</sup>OpenAI, <sup>598</sup>Google, <sup>599</sup>Meta, <sup>600</sup>OpenAI, <sup>601</sup>Google, <sup>602</sup>Meta, <sup>603</sup>OpenAI, <sup>604</sup>Google, <sup>605</sup>Meta, <sup>606</sup>OpenAI, <sup>607</sup>Google, <sup>608</sup>Meta, <sup>609</sup>OpenAI, <sup>610</sup>Google, <sup>611</sup>Meta, <sup>612</sup>OpenAI, <sup>613</sup>Google, <sup>614</sup>Meta, <sup>615</sup>OpenAI, <sup>616</sup>Google, <sup>617</sup>Meta, <sup>618</sup>OpenAI, <sup>619</sup>Google, <sup>620</sup>Meta, <sup>621</sup>OpenAI, <sup>622</sup>Google, <sup>623</sup>Meta, <sup>624</sup>OpenAI, <sup>625</sup>Google, <sup>626</sup>Meta, <sup>627</sup>OpenAI, <sup>628</sup>Google, <sup>629</sup>Meta, <sup>630</sup>OpenAI, <sup>631</sup>Google, <sup>632</sup>Meta, <sup>633</sup>OpenAI, <sup>634</sup>Google, <sup>635</sup>Meta, <sup>636</sup>OpenAI, <sup>637</sup>Google, <sup>638</sup>Meta, <sup>639</sup>OpenAI, <sup>640</sup>Google, <sup>641</sup>Meta, <sup>642</sup>OpenAI, <sup>643</sup>Google, <sup>644</sup>Meta, <sup>645</sup>OpenAI, <sup>646</sup>Google, <sup>647</sup>Meta, <sup>648</sup>OpenAI, <sup>649</sup>Google, <sup>650</sup>Meta, <sup>651</sup>OpenAI, <sup>652</sup>Google, <sup>653</sup>Meta, <sup>654</sup>OpenAI, <sup>655</sup>Google, <sup>656</sup>Meta, <sup>657</sup>OpenAI, <sup>658</sup>Google, <sup>659</sup>Meta, <sup>660</sup>OpenAI, <sup>661</sup>Google, <sup>662</sup>Meta, <sup>663</sup>OpenAI, <sup>664</sup>Google, <sup>665</sup>Meta, <sup>666</sup>OpenAI, <sup>667</sup>Google, <sup>668</sup>Meta, <sup>669</sup>OpenAI, <sup>670</sup>Google, <sup>671</sup>Meta, <sup>672</sup>OpenAI, <sup>673</sup>Google, <sup>674</sup>Meta, <sup>675</sup>OpenAI, <sup>676</sup>Google, <sup>677</sup>Meta, <sup>678</sup>OpenAI, <sup>679</sup>Google, <sup>680</sup>Meta, <sup>681</sup>OpenAI, <sup>682</sup>Google, <sup>683</sup>Meta, <sup>684</sup>OpenAI, <sup>685</sup>Google, <sup>686</sup>Meta, <sup>687</sup>OpenAI, <sup>688</sup>Google, <sup>689</sup>Meta, <sup>690</sup>OpenAI, <sup>691</sup>Google, <sup>692</sup>Meta, <sup>693</sup>OpenAI, <sup>694</sup>Google, <sup>695</sup>Meta, <sup>696</sup>OpenAI, <sup>697</sup>Google, <sup>698</sup>Meta, <sup>699</sup>OpenAI, <sup>700</sup>Google, <sup>701</sup>Meta, <sup>702</sup>OpenAI, <sup>703</sup>Google, <sup>704</sup>Meta, <sup>705</sup>OpenAI, <sup>706</sup>Google, <sup>707</sup>Meta, <sup>708</sup>OpenAI, <sup>709</sup>Google, <sup>710</sup>Meta, <sup>711</sup>OpenAI, <sup>712</sup>Google, <sup>713</sup>Meta, <sup>714</sup>OpenAI, <sup>715</sup>Google, <sup>716</sup>Meta, <sup>717</sup>OpenAI, <sup>718</sup>Google, <sup>719</sup>Meta, <sup>720</sup>OpenAI, <sup>721</sup>Google, <sup>722</sup>Meta, <sup>723</sup>OpenAI, <sup>724</sup>Google, <sup>725</sup>Meta, <sup>726</sup>OpenAI, <sup>727</sup>Google, <sup>728</sup>Meta, <sup>729</sup>OpenAI, <sup>730</sup>Google, <sup>731</sup>Meta, <sup>732</sup>OpenAI, <sup>733</sup>Google, <sup>734</sup>Meta, <sup>735</sup>OpenAI, <sup>736</sup>Google, <sup>737</sup>Meta, <sup>738</sup>OpenAI, <sup>739</sup>Google, <sup>740</sup>Meta, <sup>741</sup>OpenAI, <sup>742</sup>Google, <sup>743</sup>Meta, <sup>744</sup>OpenAI, <sup>745</sup>Google, <sup>746</sup>Meta, <sup>747</sup>OpenAI, <sup>748</sup>Google, <sup>749</sup>Meta, <sup>750</sup>OpenAI, <sup>751</sup>Google, <sup>752</sup>Meta, <sup>753</sup>OpenAI, <sup>754</sup>Google, <sup>755</sup>Meta, <sup>756</sup>OpenAI, <sup>757</sup>Google, <sup>758</sup>Meta, <sup>759</sup>OpenAI, <sup>760</sup>Google, <sup>761</sup>Meta, <sup>762</sup>OpenAI, <sup>763</sup>Google, <sup>764</sup>Meta, <sup>765</sup>OpenAI, <sup>766</sup>Google, <sup>767</sup>Meta, <sup>768</sup>OpenAI, <sup>769</sup>Google, <sup>770</sup>Meta, <sup>771</sup>OpenAI, <sup>772</sup>Google, <sup>773</sup>Meta, <sup>774</sup>OpenAI, <sup>775</sup>Google, <sup>776</sup>Meta, <sup>777</sup>OpenAI, <sup>778</sup>Google, <sup>779</sup>Meta, <sup>780</sup>OpenAI, <sup>781</sup>Google, <sup>782</sup>Meta, <sup>783</sup>OpenAI, <sup>784</sup>Google, <sup>785</sup>Meta, <sup>786</sup>OpenAI, <sup>787</sup>Google, <sup>788</sup>Meta, <sup>789</sup>OpenAI, <sup>790</sup>Google, <sup>791</sup>Meta, <sup>792</sup>OpenAI, <sup>793</sup>Google, <sup>794</sup>Meta, <sup>795</sup>OpenAI, <sup>796</sup>Google, <sup>797</sup>Meta, <sup>798</sup>OpenAI, <sup>799</sup>Google, <sup>800</sup>Meta, <sup>801</sup>OpenAI, <sup>802</sup>Google, <sup>803</sup>Meta, <sup>804</sup>OpenAI, <sup>805</sup>Google, <sup>806</sup>Meta, <sup>807</sup>OpenAI, <sup>808</sup>Google, <sup>809</sup>Meta, <sup>810</sup>OpenAI, <sup>811</sup>Google, <sup>812</sup>Meta, <sup>813</sup>OpenAI, <sup>814</sup>Google, <sup>815</sup>Meta, <sup>816</sup>OpenAI, <sup>817</sup>Google, <sup>818</sup>Meta, <sup>819</sup>OpenAI, <sup>820</sup>Google, <sup>821</sup>Meta, <sup>822</sup>OpenAI, <sup>823</sup>Google, <sup>824</sup>Meta, <sup>825</sup>OpenAI, <sup>826</sup>Google, <sup>827</sup>Meta, <sup>828</sup>OpenAI, <sup>829</sup>Google, <sup>830</sup>Meta, <sup>831</sup>OpenAI, <sup>832</sup>Google, <sup>833</sup>Meta, <sup>834</sup>OpenAI, <sup>835</sup>Google, <sup>836</sup>Meta, <sup>837</sup>OpenAI, <sup>838</sup>Google, <sup>839</sup>Meta, <sup>840</sup>OpenAI, <sup>841</sup>Google, <sup>842</sup>Meta, <sup>843</sup>OpenAI, <sup>844</sup>Google, <sup>845</sup>Meta, <sup>846</sup>OpenAI, <sup>847</sup>Google, <sup>848</sup>Meta, <sup>849</sup>OpenAI, <sup>850</sup>Google, <sup>851</sup>Meta, <sup>852</sup>OpenAI, <sup>853</sup>Google, <sup>854</sup>Meta, <sup>855</sup>OpenAI, <sup>856</sup>Google, <sup>857</sup>Meta, <sup>858</sup>OpenAI, <sup>859</sup>Google, <sup>860</sup>Meta, <sup>861</sup>OpenAI, <sup>862</sup>Google, <sup>863</sup>Meta, <sup>864</sup>OpenAI, <sup>865</sup>Google, <sup>866</sup>Meta, <sup>867</sup>OpenAI, <sup>86</sup>

# OLMo is being used to...

## Develop new architectures

NousResearch **OLMo-Bitnet-1B** like 115

Text Generation Transformers PyTorch allenai/dolma o1mo custom

License: apache-2.0

Model card

**LLaVaOLMoBitnet1B: Ternary LLM goes Multimodal!**

Jainaveen Sundaram, Ravi Iyer  
(jainaveen.sundaram, ravi@hankai.lyer)@intel.com

**OLMo-Bitnet**

OLMo-Bitnet-1B method description

**Models are in 1**

It was trained on merely a research

As separate training hyperparameters comparison ca

### Zamba: A Compact 7B SSM Hybrid Model

Paolo Glorioso Quentin Anthony Yury Tokpanov James Whittington Jonathan Pflaüt  
Adam Ibrahim Beren Millidge  
(paolo, qantenn, yury, james, jonathan, adam, beren)@zephyr.com

Zyphra  
Palo Alto, CA

### SOAP: IMPROVING AND STABILIZING SHAMPOO USING ADAM

Nikhil Vyas<sup>1</sup> Deven Morwani<sup>1</sup> Rosie Zhao<sup>1</sup> Itai Shapira<sup>1</sup>  
Harvard University Harvard University Harvard University Harvard University

David Brandfonbrenner<sup>1</sup> Lucas Janson<sup>1</sup> Sham Kakade<sup>1</sup>  
Kempner Institute at Harvard University Kempner Institute at Harvard University Kempner Institute at Harvard University

#### ABSTRACT

There is growing evidence of the effectiveness of Shampoo, a higher-order preconditioning method, over Adam in deep learning optimization tasks. However, Shampoo's drawbacks include additional hyperparameters and computational overhead when compared to Adam, which only updates running averages of first- and second-moment quantities. This work establishes a formal connection between Shampoo (implemented with the  $1/2$  power) and Adafactor — a memory-efficient approximation of Adam — showing that Shampoo is equivalent to running Adafactor in the eigenbasis of Shampoo's preconditioner. This insight leads to the design of a simpler and computationally efficient algorithm: Shampoo<sup>2</sup> with Adam in the Preconditioner's eigenbasis (SOAP). With regards to improving Shampoo's computational efficiency, the most straightforward approach would be to simply compute Shampoo's eigendecomposition less frequently. Unfortunately, as our empirical results show, this leads to performance degradation that worsens with this frequency. SOAP mitigates this degradation by continually updating the running average of the second moment, just as Adam does, but in the current (slowly changing) coordinate basis. Furthermore, since SOAP is equivalent to running Adam in a rotated space, it introduces only one additional hyperparameter (the preconditioning frequency) compared to Adam. We empirically evaluate SOAP on language model pre-training with 360m and 660m sized models. In the large batch regime, SOAP reduces the number of iterations by over 40% and wall clock time by over 35% compared to AdamW, with approximately 20% improvements in both metrics compared to Shampoo. An implementation of SOAP is available at <https://github.com/nikhilvyas/soap>.

## Study how models acquire/forget knowledge during training

### Adaptive Pre-training Data Detection for Large Language Models via Surprising Tokens

Angi Zhang

Anwger, Assemle, Acee  
Understanding How Transformers Answer Multiple Choice Questions

Sarah Wiegrefe<sup>1,2\*</sup> Oyvind Tafjord<sup>1</sup> Yonatan Belinkov<sup>1</sup>  
Hannaneh Hajishiraf<sup>1,2\*</sup> Ashish Sabharwal<sup>2</sup>

<sup>1</sup>Allen Institute for AI, <sup>2</sup>University of Washington, <sup>3</sup>Technion  
wiegrefe@allenai.com

#### Abstract

Multiple-choice question answering is a key component of performance in language models that is tested by its benchmarks. However, recent evidence that models can have quite a strong bias, particularly when the task is verified slightly (such as by a shell choice order). In this work we ask if careful models perform formally to employ vocabulary projection and patching methods to localize key tokens that encode relevant information for the correct answer. We find that pre-training a specific answer symbol in  $v$  space to a single middle layer, and spec multi-head self-attention mechanism that subsequent layers increase the of the predicted answer symbol in  $v$  space, and that this probability is associated with a sparse set of attention unique nodes. We additionally used synthetic tasks on downstream question to pinpoint when a model has learned MQA, and show that we can separate answer symbol tokens in  $v$  space as a property of models unable to format MQA tasks.

#### Abstract

Despite the proven utility of large language models (LLMs) in real-world applications, a lack of understanding regarding how they leverage their large-scale pretraining to achieve such capabilities. In this investigation the interplay between generalization and memorization in pretraining LLMs through a comprehensive n-gram analysis. Our experiments focus on generalization vs. memorization: Tracing Language Models' Capabilities Back to Pretraining Data

Antonis Antoniadou<sup>1</sup> Xinyi Wang<sup>2</sup>

#### Abstract

Recent work on evaluating the diversity of text generated by LLMs has focused on word-level features. Here we offer an analysis of syntactic features to characterize general repetition in models, beyond frequent n-grams. Specifically, we define syntactic templates and show that models tend to produce templated text in downstream tasks at a higher rate than what is found in human-reference texts. We find that most (76%) templates in model-generated text can be found in pre-training data (compared to only 35% of human-authored text), and are not over-represented during fine-tuning processes such as RLHF. This connection to the pre-training data allows us to analyze syntactic templates in models where we do not have the pre-training data. We also find that templates as features are able to differentiate between models, tasks, and domains, and are useful for qualitatively evaluating common model constructions. Finally, we demonstrate the use of templates as a useful tool for analyzing style memorization of training data in LLMs.

### Generalization vs. Memorization: Tracing Language Models' Capabilities Back to Pretraining Data

### How Do Large Language Models Acquire Factual Knowledge During Pretraining?

Hyeon Chang<sup>1</sup> Jinho Park<sup>1</sup> Seungyeon Yee<sup>1</sup> Seokyeung Yoo<sup>2</sup>

### Detection and Measurement of Syntactic Templates in Generated Text

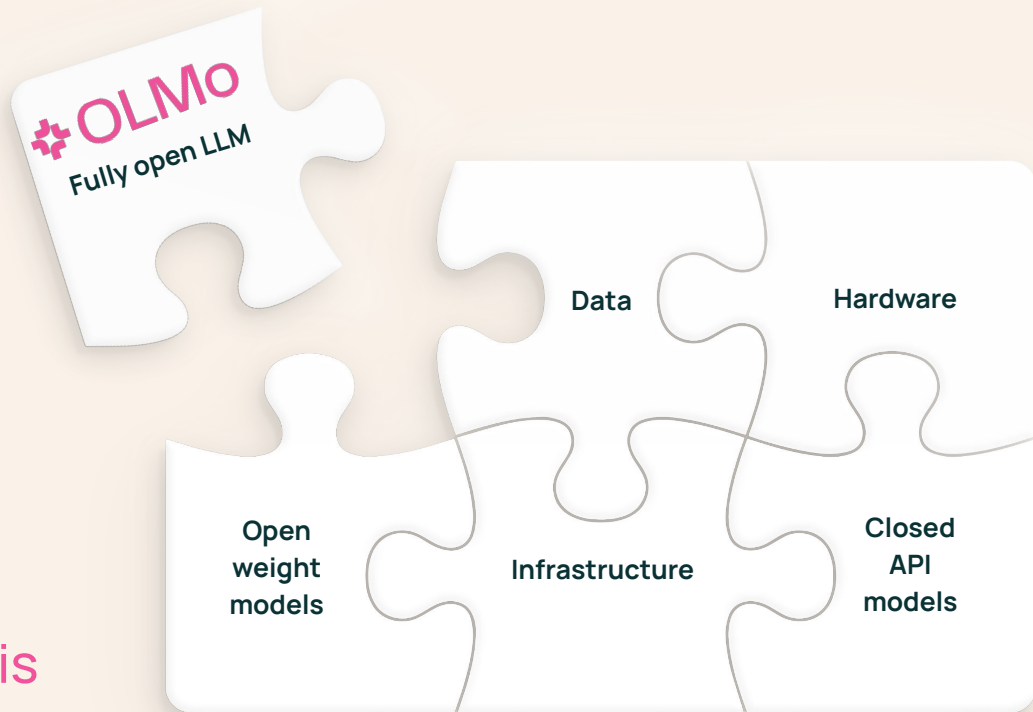
Chantall Shah<sup>1</sup> Yanal Elazar<sup>1,3</sup> Junyi Jessy Li<sup>1</sup> Byron C. Wallace<sup>1</sup>  
<sup>1</sup>Northwestern University, <sup>2</sup>Allen Institute for AI, <sup>3</sup>University of Washington, <sup>4</sup>The University of Texas at Austin  
(shah.c, wallace@northwestern.edu)

#### Abstract

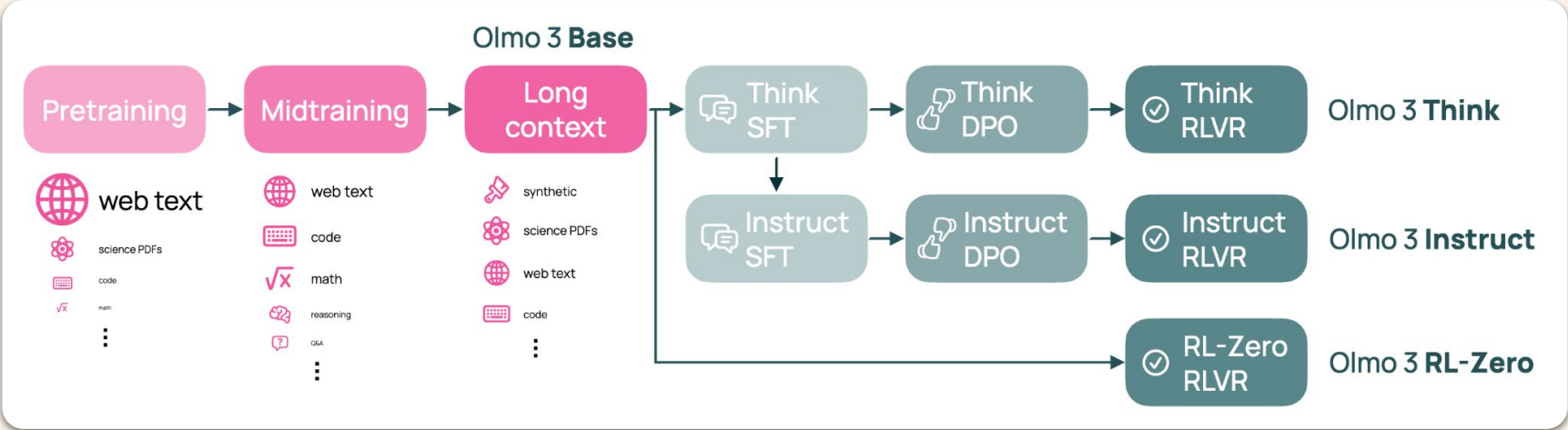
Recent work on evaluating the diversity of text generated by LLMs has focused on word-level features. Here we offer an analysis of syntactic features to characterize general repetition in models, beyond frequent n-grams. Specifically, we define syntactic templates and show that models tend to produce templated text in downstream tasks at a higher rate than what is found in human-reference texts. We find that most (76%) templates in model-generated text can be found in pre-training data (compared to only 35% of human-authored text), and are not over-represented during fine-tuning processes such as RLHF. This connection to the pre-training data allows us to analyze syntactic templates in models where we do not have the pre-training data. We also find that templates as features are able to differentiate between models, tasks, and domains, and are useful for qualitatively evaluating common model constructions. Finally, we demonstrate the use of templates as a useful tool for analyzing style memorization of training data in LLMs.

# What does fully open mean?

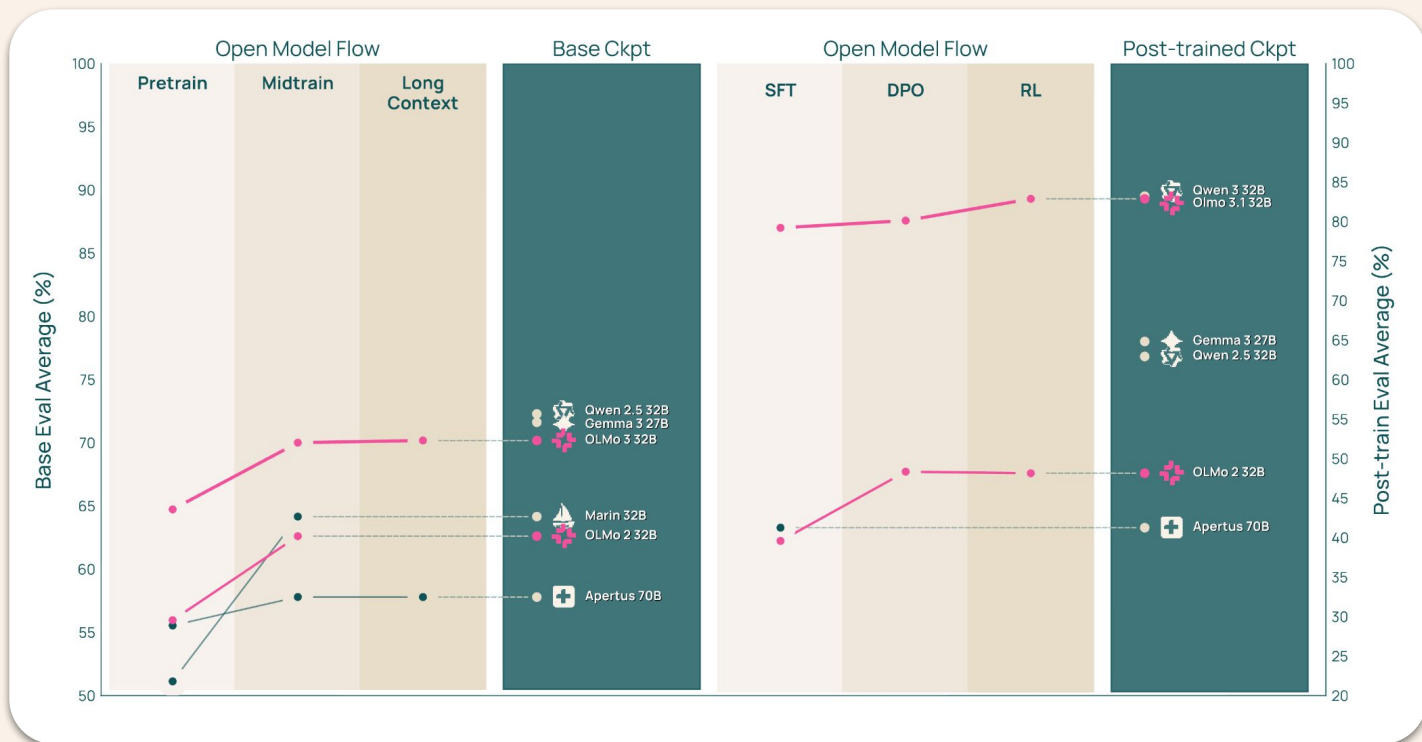
- Model **weights**
- Intermediate **checkpoints**
- Detailed **recipes**
- All the **data**
- **Code** to reproduce
- **Documentation** and **analysis**



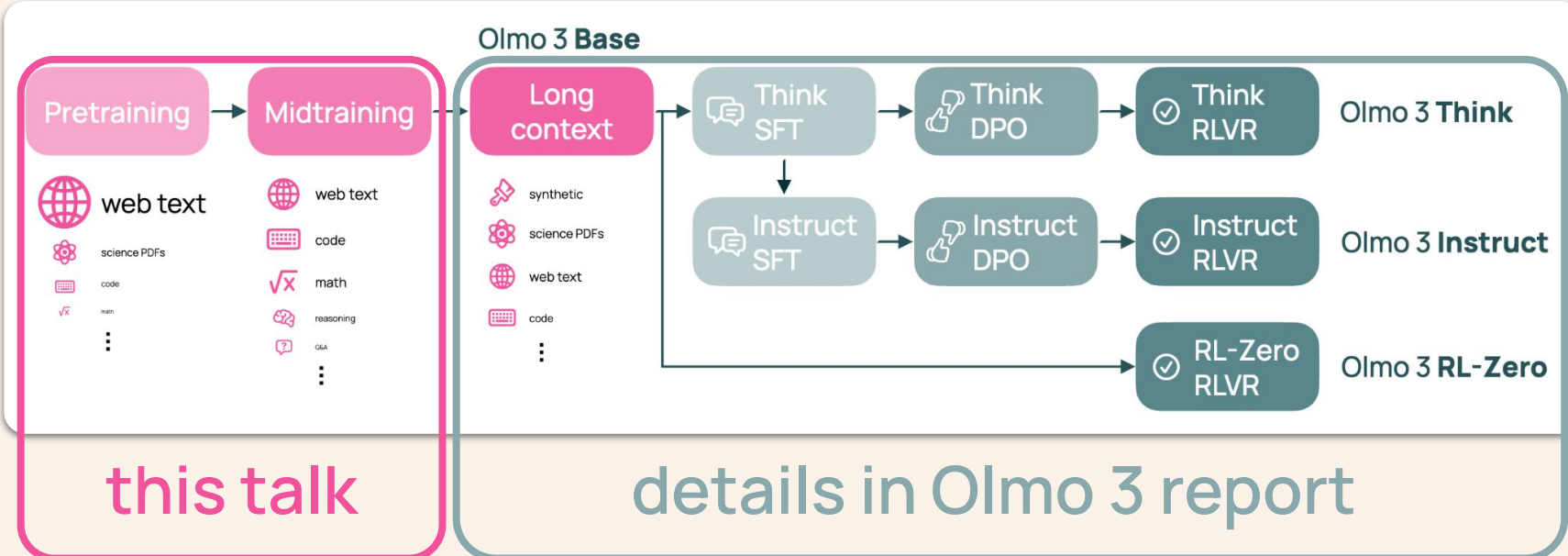
# Olmo 3 Model Flow



# Olmo 3 Model Flow



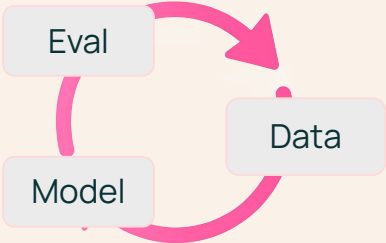
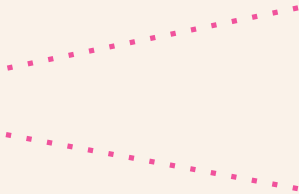
# Olmo 3 Model Flow



this talk

details in Olmo 3 report

# Olmo 3 Model Flow



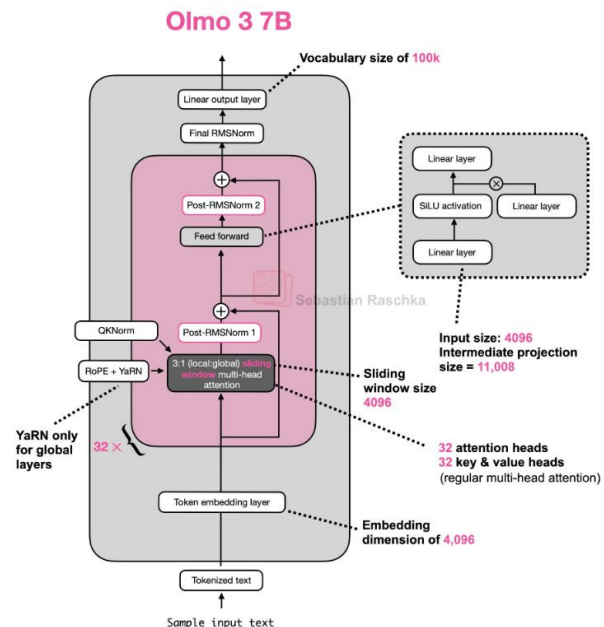




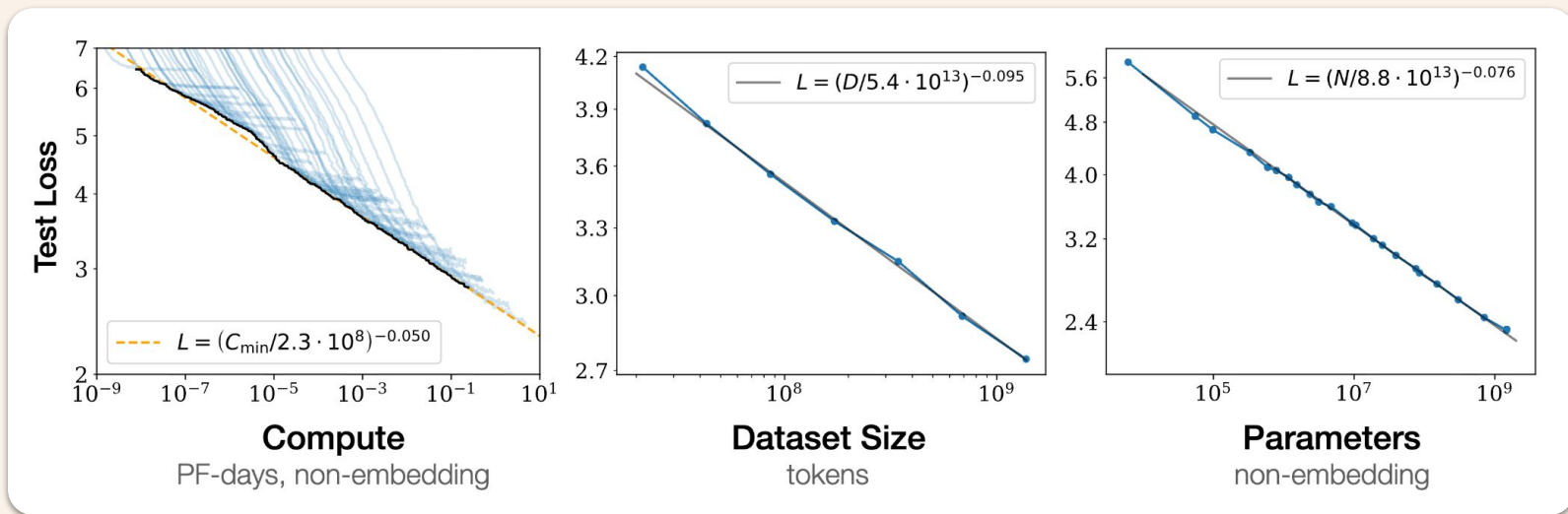
# Background

# Background: Architecture Changes

Layers	<b>32 / 64</b>	Gradient clipping	1.0
Hidden size ( $d_{\text{model}}$ )	<b>4096 / 5120</b>	Z-loss weight	$10^{-5}$
Q heads	<b>32 / 40</b>	Weight decay on embeddings	No
KV heads	<b>32 / 8</b>	Sliding window attention	3/4 of layers; 4,096 tokens
Activation	SwiGLU	RoPE scaling	YaRN on full attn. layers
QKV normalization	QK-Norm	RoPE $\theta$	$5 \cdot 10^5$
Layer norm	RMSNorm	Layer norm applied to	Outputs



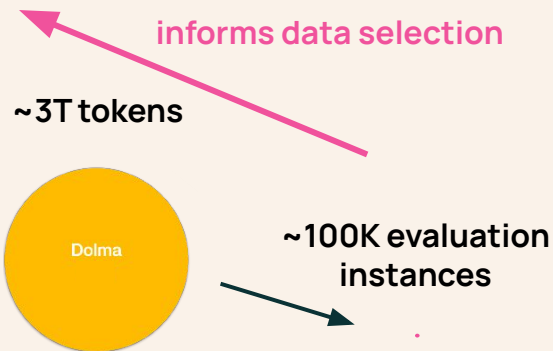
# Background: Scaling Pretrained Models

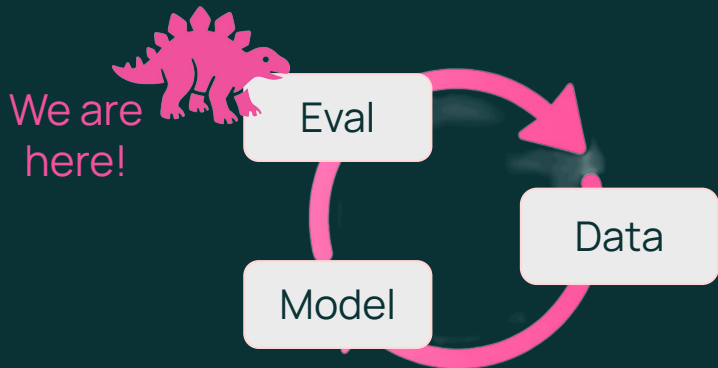


Rough estimate:  
text in ~30M books in the US Library of  
Congress  $\approx$  between 1/400 and 1/24 PB

Common Crawl  
~1 PB

## Background: Data





## Signal and Noise: A Framework for Reducing Uncertainty in Language Model Evaluation

David Heineman<sup>1\*</sup> Valentin Hofmann<sup>1,2\*</sup> Ian Magnusson<sup>1,2\*</sup> Yuling Gu<sup>1\*</sup>  
Noah A. Smith<sup>1,2\*</sup> Hannaneh Hajishirzi<sup>1,2\*</sup> Kyle Lo<sup>1</sup> Jesse Dodge<sup>1</sup>

<sup>1</sup>Allen Institute for Artificial Intelligence

<sup>2</sup>Paul G. Allen School of Computer Science & Engineering, University of Washington  
contact: davidh@allenai.org

### Abstract

Developing large language models is expensive and involves making decisions with small experiments, typically by evaluating on large, multi-task evaluation suites. In this work, we analyze specific properties which make a benchmark more reliable for such decisions, and interventions to design higher-quality evaluation benchmarks. We introduce two key metrics that show differences in current benchmarks: *signal*, a benchmark's ability to separate better models from worse models, and *noise*, a benchmark's sensitivity to random variability between training steps. We demonstrate that benchmarks with a better *signal-to-noise* ratio are more reliable when making decisions at small scale, and those with less *noise* have lower scaling law prediction error. These results suggest that improving *signal* or *noise* will lead to more useful benchmarks, so we introduce three interventions designed to directly affect *signal* or *noise*. For example, we propose that switching to a metric that has better *signal* and *noise* (e.g., perplexity rather than accuracy) leads to better reliability and improved scaling law error. We also find that filtering noisy subtasks, to improve an aggregate *signal-to-noise* ratio, leads to more reliable multi-task evaluations. We also find that averaging the output of a model's intermediate checkpoints to reduce *noise* leads to consistent improvements. We conclude by recommending that those creating new benchmarks, or selecting which existing benchmarks to use, aim for high *signal* and low *noise*. We use 30 benchmarks for these experiments, and 375 open-weight language models from 60M to 32B parameters, resulting in a new, publicly available dataset of 900K evaluation benchmark results, totaling 200M instances.

[github.com/allenai/signal-and-noise](https://github.com/allenai/signal-and-noise) [huggingface.co/datasets/allenai/signal-and-noise](https://huggingface.co/datasets/allenai/signal-and-noise)

### 1 Introduction

Language model development is expensive. During the development process, researchers need to make decisions such as what architecture to use, what training methods to employ, and what data to

# How to make sense of the wall of results?

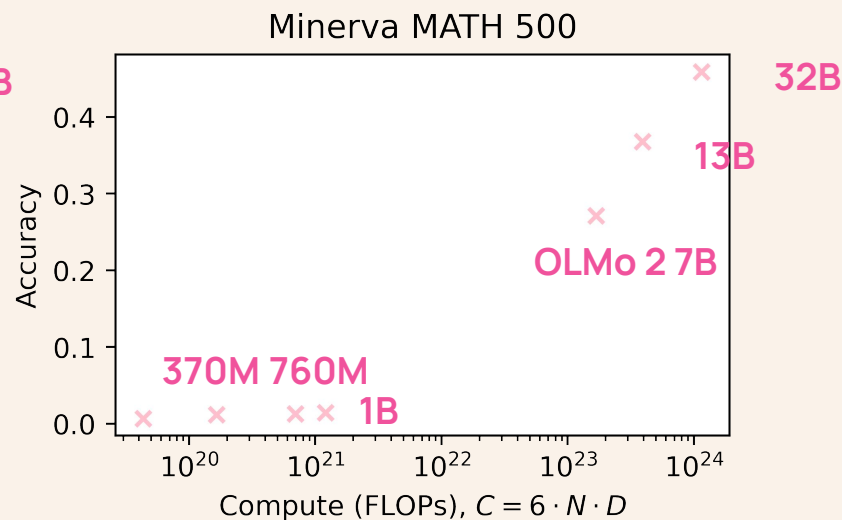
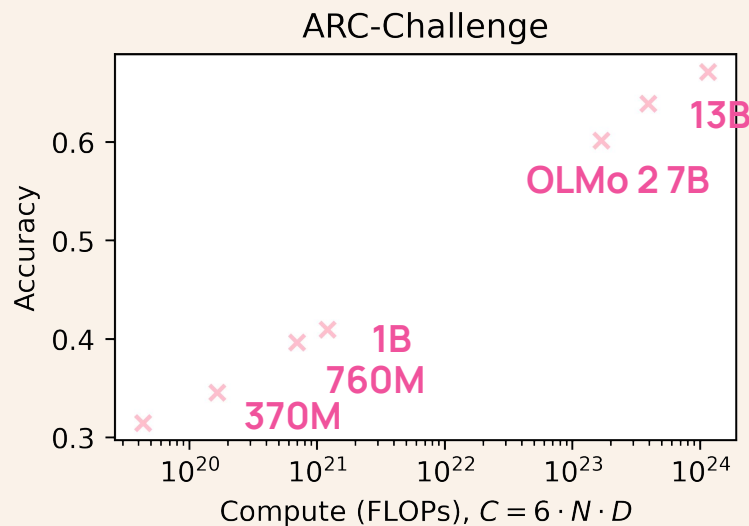
	Fully-open Models						Open-weight Models					
	Olmo 3 32B	Marin 32B	Apertus 70B	Gaperon 24B	LLM 360 K2V270B	OLMO 2 32B	Owen 2.5 32B	Gemma 3 27B	Mistral 3.1 24B	Seed 36B	Gemma 2 27B	Llama 3.1 70B
<b>OlmoBaseEval Math</b>	61.9	49.3	39.7	20.7	46.2	53.9	64.7	63.2	59.5	15.3	57.5	62.0
GSMsk	80.6	69.1	63.0	33.3	66.7	77.6	81.1	81.3	79.3	26.9	76.3	81.2
GSM Symbolic	61.2	42.0	38.6	14.5	44.4	53.1	56.2	61.2	59.1	10.3	57.3	64.6
MATH	43.8	36.8	17.4	14.2	27.4	31.0	56.7	47.0	40.1	8.7	38.8	40.2
<b>OlmoBaseEval Code</b>	39.7	30.8	23.3	19.4	35.2	20.5	48.3	41.6	42.4	54.9	41.0	36.3
BigCodeBench	43.7	34.5	24.0	17.0	39.8	22.2	48.1	44.0	46.4	50.7	43.4	43.4
HumanEval	65.8	52.3	32.5	31.2	51.2	29.4	65.6	62.1	65.5	71.3	57.5	57.4
DeepSeek LeetCode	2.0	1.3	1.2	0.0	2.3	0.8	8.0	5.8	0.1	13.0	4.7	0.2
DS 1000	29.4	26.3	17.8	11.0	25.4	20.4	43.3	34.3	36.3	44.0	29.7	29.5
MBPP	59.6	52.1	37.6	36.7	53.5	37.1	69.8	60.0	61.9	72.0	61.7	55.5
MultiPL HumanEval	36.0	18.5	18.4	13.0	31.3	10.5	49.7	37.7	39.0	69.2	40.3	32.2
MultiPL MBPPP	41.5	30.5	31.3	26.5	42.8	23.2	53.6	47.2	47.7	63.8	49.7	35.9
<b>OlmoBaseEval MC STEM</b>	74.5	75.9	70.0	56.2	75.6	75.3	82.2	80.2	81.5	83.4	75.6	80.1
ARC MC	94.7	93.4	90.7	72.7	93.0	94.4	97.0	95.8	96.2	97.3	94.1	95.2
MMLU STEM	70.8	68.4	57.8	45.3	64.7	64.7	79.7	74.9	76.1	82.8	65.8	70.0
MedMCQA MC	57.6	61.8	55.9	42.6	63.7	60.2	68.8	64.7	68.8	69.6	61.8	67.8
MedQA MC	53.8	60.8	52.4	35.4	61.4	62.2	68.4	68.7	70.4	70.1	61.0	72.3
SciQ MC	95.5	95.1	93.3	84.9	95.3	95.1	97.1	96.8	96.3	97.1	95.1	95.4
<b>OlmoBaseEval MC Non-STEM</b>	85.6	84.5	78.5	64.1	83.5	84.2	89.3	86.7	87.9	89.0	83.2	86.1
MMLU Humanities	78.3	78.9	74.1	56.7	79.3	79.7	85.0	80.5	82.7	85.7	79.3	83.4
MMLU Social Sci.	84.0	83.7	79.2	58.9	84.9	84.5	88.4	86.2	88.6	90.1	85.8	87.4
MMLU Other	75.1	75.4	70.1	55.4	76.3	75.6	81.2	80.2	81.9	82.4	76.9	79.4
CSQA MC	82.3	80.1	76.9	60.6	78.6	81.2	89.9	79.0	80.5	81.1	78.1	79.0
PiQA MC	85.6	90.5	79.0	72.0	87.3	87.7	93.3	90.3	91.0	92.5	89.0	91.5
SocialIQa MC	83.9	82.4	79.3	71.3	81.2	82.3	86.6	81.2	81.0	84.9	81.0	83.5
CoQA Gen2MC MC	96.4	93.9	87.5	67.3	92.0	94.4	96.8	95.8	94.9	96.9	94.3	95.1
DROP Gen2MC MC	87.2	71.0	56.5	48.0	64.8	68.6	86.6	84.6	86.5	90.1	66.6	70.3
Jeopardy Gen2MC MC	92.3	95.3	93.2	77.0	95.3	96.6	97.0	95.9	97.2	96.2	92.0	97.1
NaturalQs Gen2MC MC	78.0	81.0	71.9	47.5	82.4	78.6	79.9	82.0	84.6	81.4	74.5	82.4
SQuAD Gen2MC MC	98.2	97.6	95.7	90.0	96.7	97.4	97.9	97.7	97.9	98.1	97.5	97.7
<b>OlmoBaseEval GenQA</b>	79.8	80.3	75.0	65.3	77.1	79.1	68.5	73.5	78.0	76.0	72.9	81.6
HellaSwag RC	84.8	87.2	84.5	75.2	87.6	87.5	86.3	86.0	86.2	84.8	86.7	88.4
Winogrande RC	90.3	90.5	87.7	80.3	88.9	89.4	87.5	91.3	90.8	89.3	90.8	91.7
Lambada	75.7	76.7	74.8	58.3	76.8	77.0	76.2	77.5	79.3	76.1	76.9	79.6
Basic Skills	93.5	91.1	87.5	83.2	90.6	88.7	94.2	94.9	91.9	96.0	93.2	92.4
DROP	80.9	76.5	56.3	59.4	69.7	76.3	53.7	75.9	74.9	76.1	73.2	78.3
Jeopardy	75.3	80.5	77.2	58.9	79.8	79.1	74.0	82.1	80.3	77.4	80.7	84.0
NaturalQs	49.0	55.1	43.1	33.5	47.6	51.4	39.3	49.2	45.1	30.7	47.1	53.1
SQuAD	94.5	94.4	90.7	89.3	91.2	94.0	64.9	92.4	92.6	89.1	93.0	92.9
CoQA	74.1	70.7	72.8	49.8	61.5	68.7	40.4	12.4	61.1	64.4	14.9	73.9
<b>OlmoBaseEval HeldOut</b>												
LBPP	21.8	17.3	8.1	4.3	13.4	8.2	40.3	17.7	30.3	42.6	19.7	11.8
BBH	77.6	70.1	58.8	36.6	73.2	64.6	81.1	77.4	81.4	85.0	74.8	80.8
MMLU Pro MC	49.7	48.1	39.6	21.3	45.3	46.9	61.1	53.1	58.9	62.2	47.6	50.4
Deepmind Math	29.6	26.7	20.1	28.3	32.5	22.0	40.7	30.4	35.3	31.3	27.6	40.2

**Table 2 Results comparing Olmo 3 Base 32B to other base models using the OlmoBaseEval Main suite** (details in Section §3.3). OLMO 3 was not evaluated on held-out benchmarks prior to release.

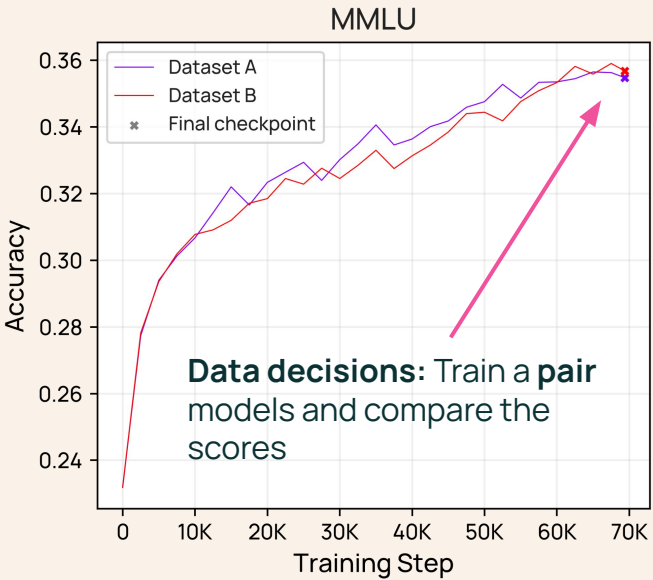
We evaluate many families of abilities (QA, Math, Coding, ...)

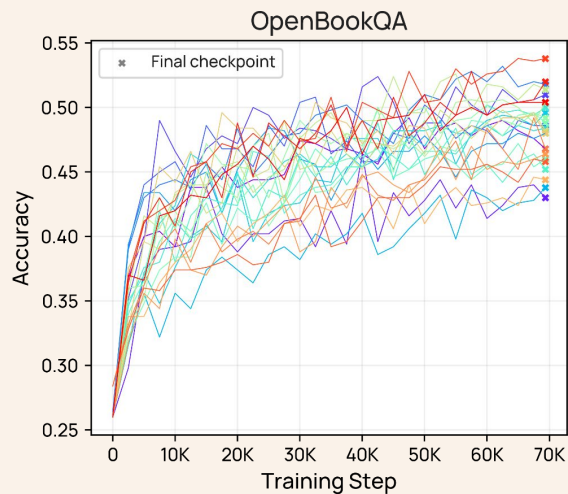
HellaSwag CSQA DROP TriviaQA DS-1000 BigCodeBench  
CoQA Lambada PiQA ARC-C MedMCQA HumanEval MBPP  
WinoGrande BoolQ ARC-E MMLU GSM8K CruxEval  
SocialIQA Jeopardy SQuAD Minerva MATH

## ... but how to get **signal** for small-scale decisions?



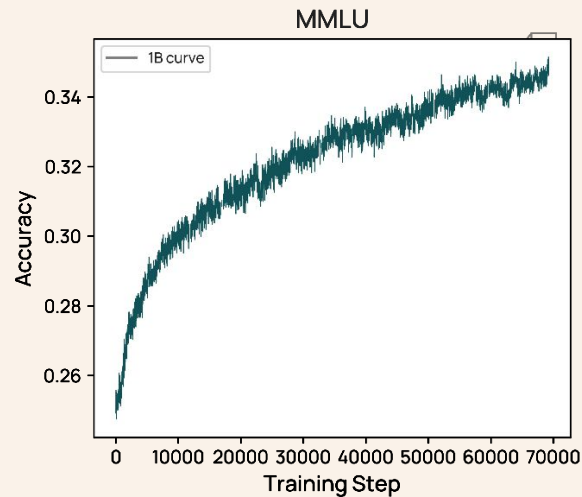
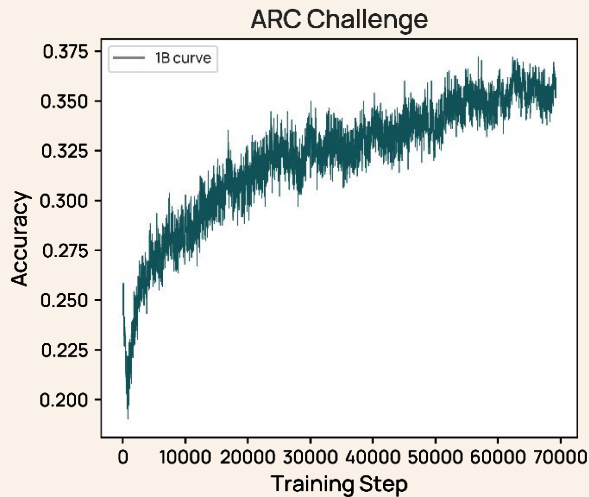
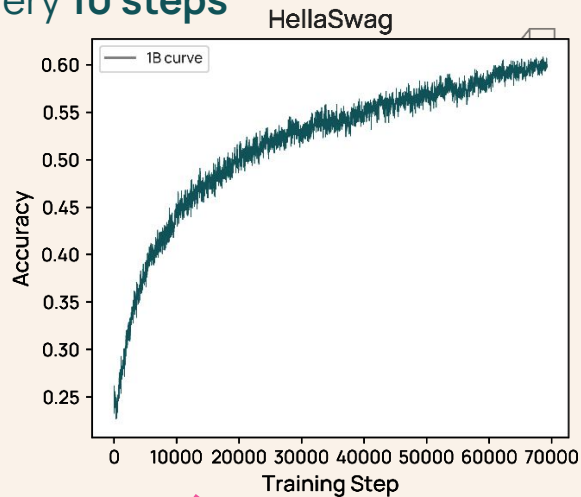
# Making decisions in pretraining == extrapolating eval



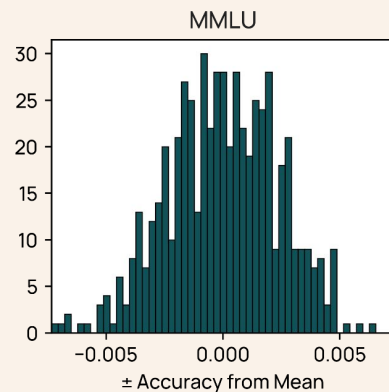
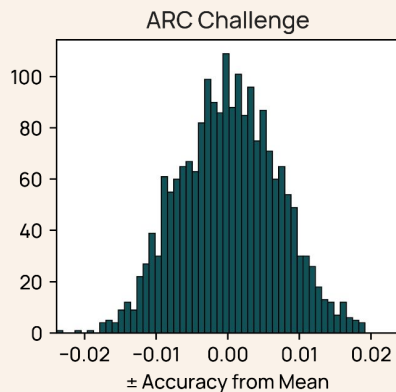
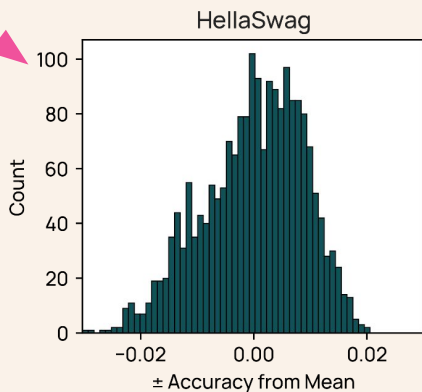


Why do so many predictions fail -  
but some don't?

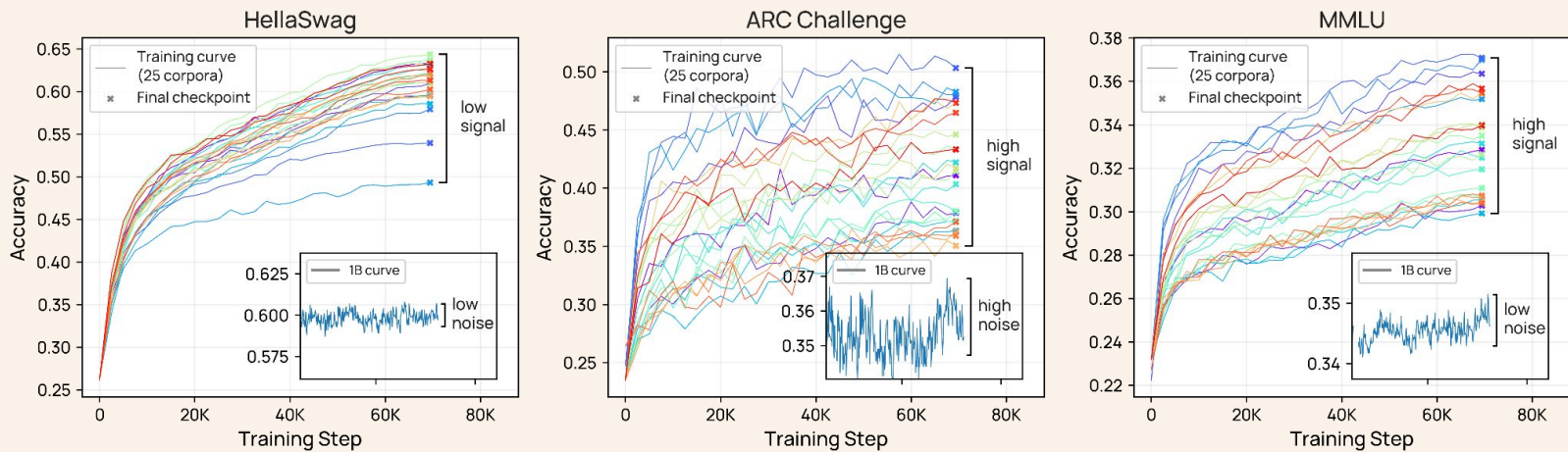
# 1B-5xC + eval every 10 steps

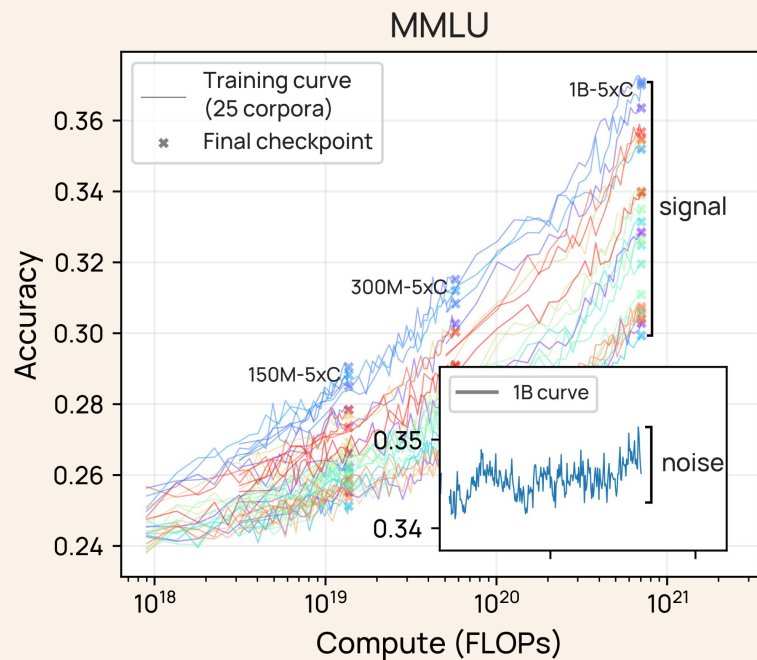


Final 20% of checkpoints



# Inter-checkpoint variance is not the whole story! We need to measure both **signal** and **noise**





**Signal** (max difference between model scores):

$$\text{Rel. Dispersion}(M) = \max_{j,k} |m_j - m_k| / \bar{m}$$

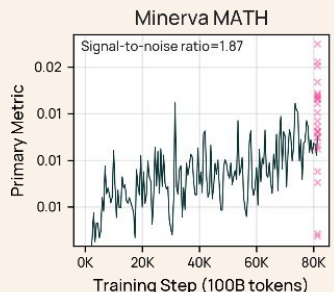
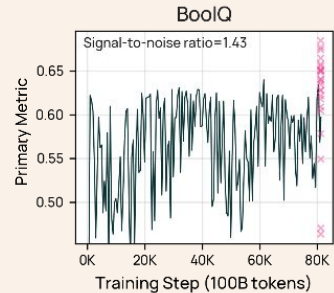
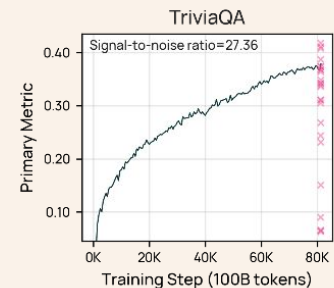
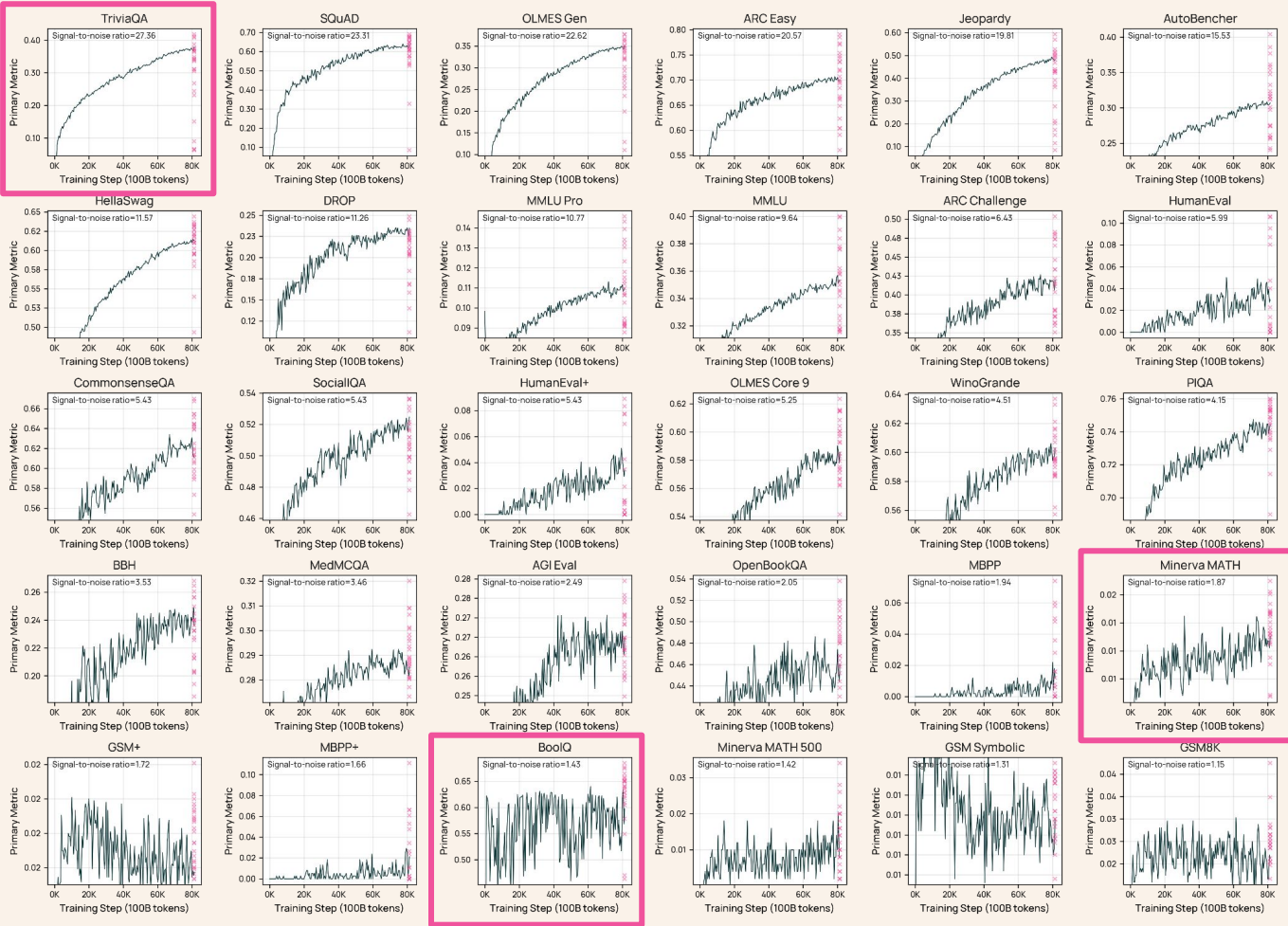
**Noise** (ckpt-to-ckpt noise using last n steps):

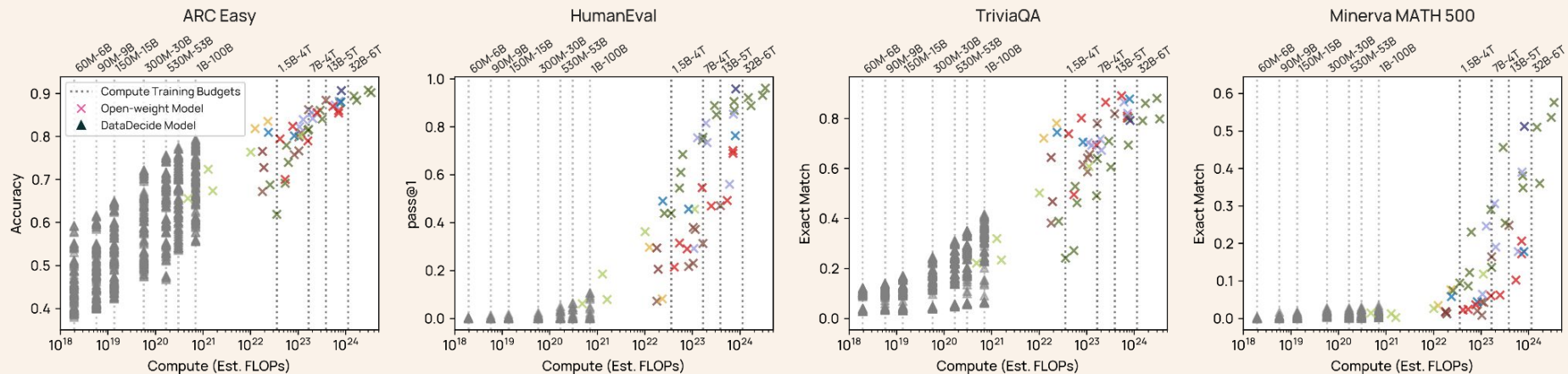
$$\text{Rel. Std.}(m) = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (m_i - \bar{m})^2 / \bar{m}}$$

Signal-to-Noise Ratio =

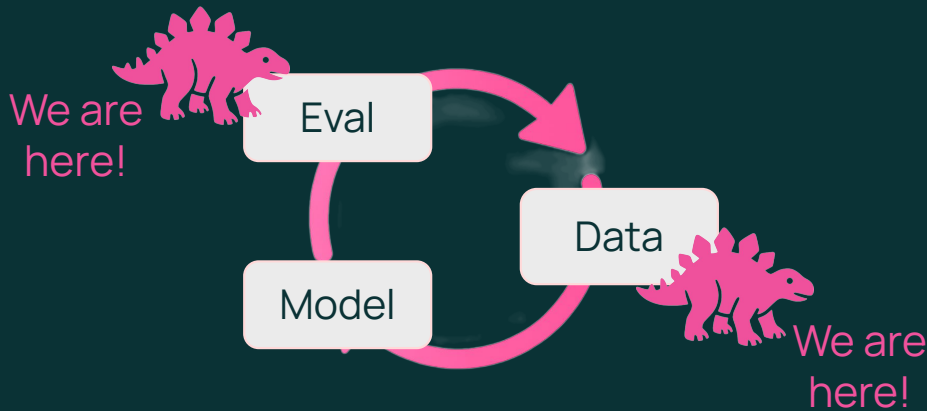
$$\frac{\text{Rel. Dispersion}(\text{final train checkpoint})}{\text{Rel. Std.}(\text{final } n \text{ train checkpoints})}$$

← SNR at the 1B scale





The signal-to-noise ratio is specifically useful as **signal of benchmarks changes** for large train compute budgets



# Olmix: A Framework for Data Mixing Throughout LM Development

Mayee F. Chen<sup>1,2</sup> Tyler Murray<sup>1</sup> David Heineman<sup>1</sup> Matt Jordan<sup>1</sup> Hannaneh Hajishirzi<sup>1,3</sup>  
 Christopher Ré<sup>2</sup> Luca Soldaini<sup>1</sup> Kyle Lo<sup>1,3</sup>

<sup>1</sup>Allen Institute for AI <sup>2</sup>Stanford University <sup>3</sup>University of Washington

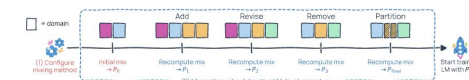
Code: [Olmix](#) Data: [Olmix](#) Contact: [mfchen@cs.stanford.edu](mailto:mfchen@cs.stanford.edu) [{lucas,kyle}@allenai.org](mailto:{lucas,kyle}@allenai.org)

## Abstract



Data mixing—determining the ratios of data from different domains—is a first-order concern for training language models (LMs). While existing mixing methods show promise, they fall short when applied during real-world LM development. We present OLMIX, a framework that addresses two such challenges. First, the configuration space for developing a mixing method is not well understood—design choices across existing methods lack justification or consensus and overlook practical issues like data constraints. We conduct a comprehensive empirical study of this space, identifying which design choices lead to a strong mixing method. Second, in practice, the domain set evolves throughout LM development as datasets are added, removed, partitioned, and revised—a problem setting largely unaddressed by existing works, which assume fixed domains. We study how to efficiently recompute the mixture after the domain set is updated, leveraging information from past mixtures. We introduce mixture reuse, a mechanism that reuses existing ratios and recomputes ratios only for domains affected by the update. Over a sequence of five domain-set updates mirroring real-world LM development, mixture reuse matches the performance of fully recomputing the mix after each update with 74% less compute and improves over training without mixing by 11.6% on downstream tasks.

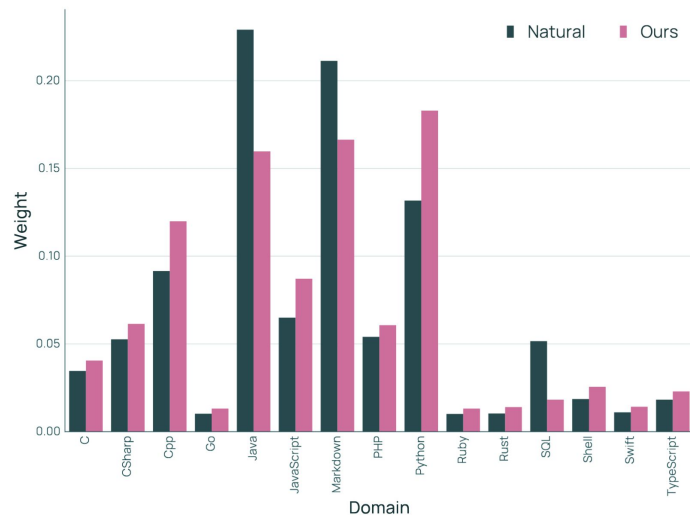
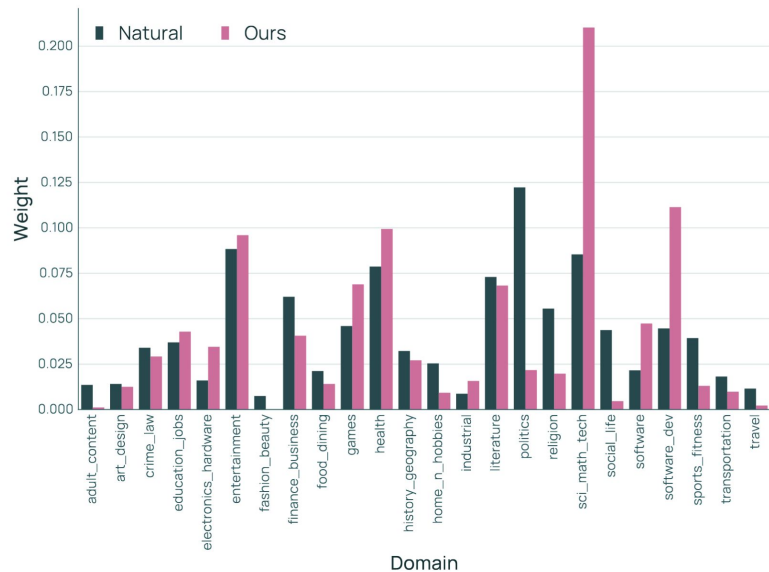
## 1 Introduction



**Figure 1** Two problems with data mixing encountered during LM development: (1) How to best configure your mixing method? (2) How to efficiently mix under evolving domain sets?

Modern language models (LMs) are trained on datasets composed of many domains, such as web text, code, and PDFs. The composition of these domains is crucial for strong downstream performance, making data mixing a first-order component of LM development (Grazziotin et al., 2024; Chen et al., 2024; Ohno et al., 2025). However, finding a good mix is non-trivial: practitioners often resort to manual weight tuning or exhaustive search, which can require many training runs—possibly thousands of GPU hours—to assess performance. This has resulted in a growing literature on data mixing methods that aim to find strong mixtures systematically with less compute (Liu et al., 2023; Fan et al., 2024; Chen et al., 2025). Many mixing methods that achieve promising results follow a common *offline mixing schema* (Liu et al., 2023b; Ye et al., 2025) that consists of three steps: 1. train a set of smaller proxy models on different mixtures (a “swarm”), 2.

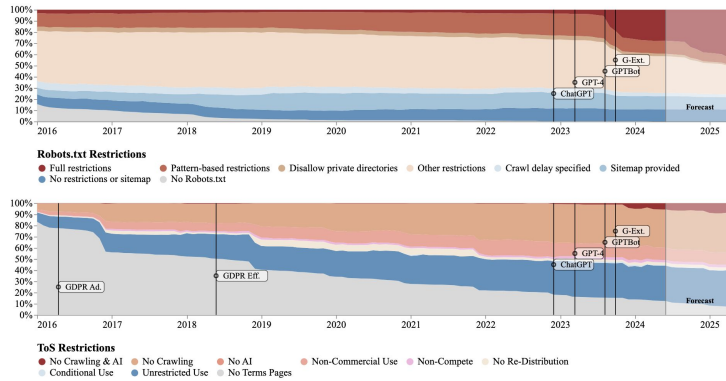
# What data mixing?



# Why data mixing?

## Token scale

### Consent in Crisis (Neurips 2024)



## Proportion / Ratios

### Gopher (2021)

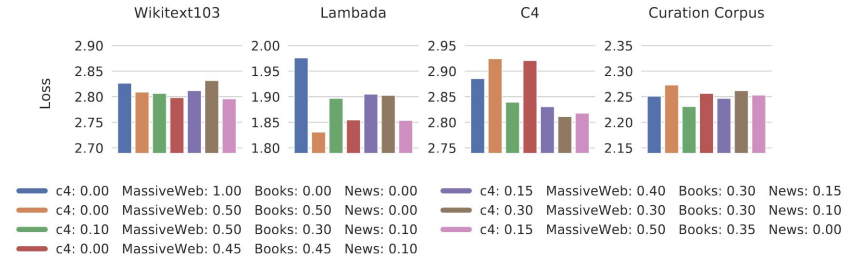
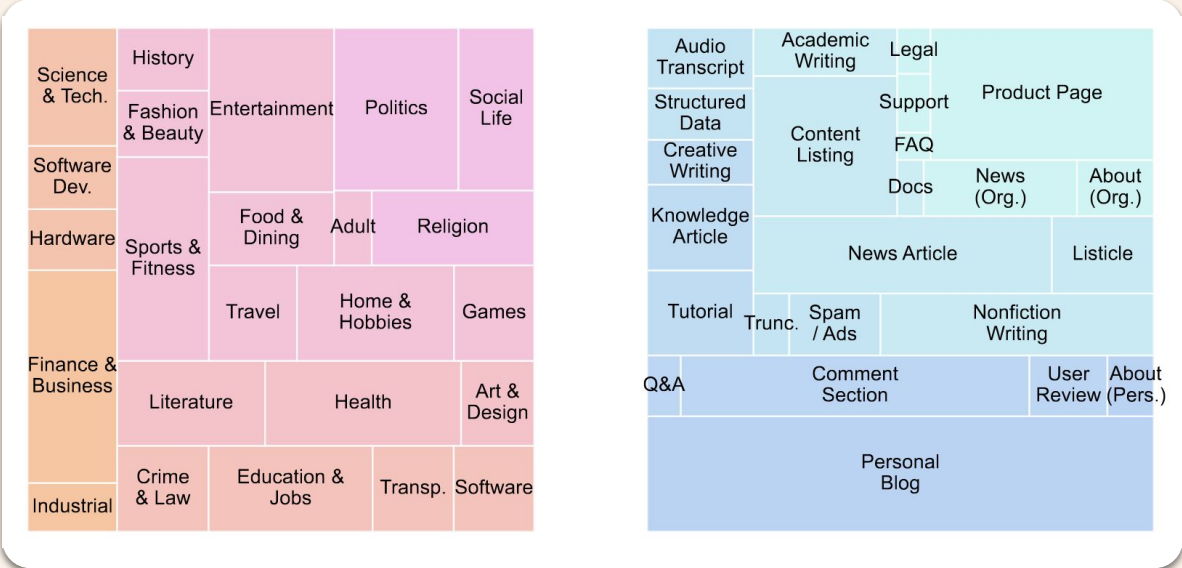


Figure A4 | Downstream performance for different *MassiveText* subset sampling weights. The configuration (in green) with 10% C4, 50% *MassiveWeb*, 30% Books, and 10% News performs well across all tasks and achieves the best performance on Curation Corpus—we therefore choose those sampling weights in our main *Gopher* training experiments.

# Problem: Search over data mixtures is combinatorial



# How data mixing?

## REGMIX: DATA MIXTURE AS REGRESSION FOR LANGUAGE MODEL PRE-TRAINING

**Qian Liu<sup>1\*</sup> Xiaosen Zheng<sup>2\*</sup> Niklas Muennighoff<sup>3,4</sup> Guangtao Zeng<sup>5</sup>**  
**Longxu Dou<sup>1</sup> Tianyu Pang<sup>1</sup> Jing Jiang<sup>2</sup> Min Lin<sup>1</sup>**  
<sup>1</sup>Sea AI Lab <sup>2</sup>SMU <sup>3</sup>Contextual AI <sup>4</sup>Stanford University <sup>5</sup>SUTD  
liuqian.sea@gmail.com; xszheng.2020@phdcs.smu.edu.sg

## BiMIX: BIVARIATE DATA MIXING LAW FOR LANGUAGE MODEL PRETRAINING

**Ce Ge & Zhijian Ma**  
Alibaba Group  
Beijing, China  
{gece.gc, zhijian.mzj}@alibaba-inc.com

**Dayuan Chen**  
Alibaba Group  
Hangzhou, China  
daoyuanchen.cdy@alibaba-inc.com

**Yaliang Li\* & Bolin Ding**  
Alibaba Group  
Bellevue, USA  
{yaliang.li, bolin.ding}@alibaba-inc.com

## Data Mixing Laws: Optimizing Data Mixtures by Predicting Language Modeling Performance

**Jiasheng Ye<sup>1,\*</sup> Peiju Liu<sup>1,\*</sup> Tianxiang Sun<sup>1</sup> Jun Zhan<sup>1</sup> Yunhua Zhou<sup>2,†</sup> Xipeng Qiu<sup>1,†</sup>**  
{jsye23, pjliu23}@m.fudan.edu.cn zhouyunhua@pjlab.org.cn xpqiu@fudan.edu.cn  
<sup>1</sup>Fudan University <sup>2</sup>Shanghai AI Laboratory

---

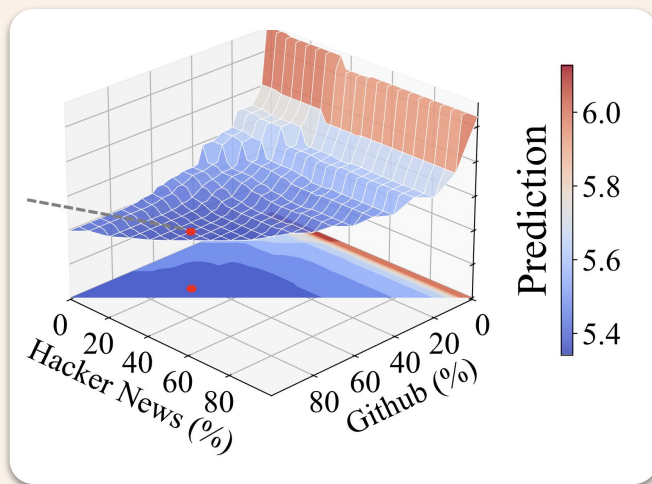
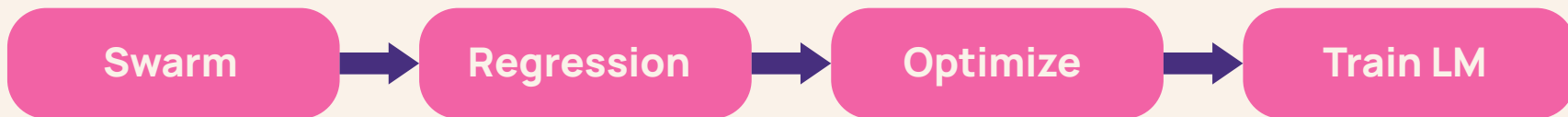
## Nemotron-CLIMB: CLustering-based Iterative Data Mixture Bootstrapping for Language Model Pre-training

---

**Shizhe Diao<sup>1</sup>, Yu Yang<sup>1</sup>, Yonggan Fu<sup>1</sup>, Xin Dong<sup>1</sup>, Dan Su<sup>1</sup>, Markus Kliegl<sup>1</sup>, Zijia Chen<sup>1</sup>,  
Peter Belcak<sup>1</sup>, Yoshi Suhara<sup>1</sup>, Hongxu Yin<sup>1</sup>, Mostafa Patwary<sup>1</sup>, Yingyan (Celine) Lin<sup>2</sup>,  
Jan Kautz<sup>1</sup>, Pavlo Molchanov<sup>1</sup>**

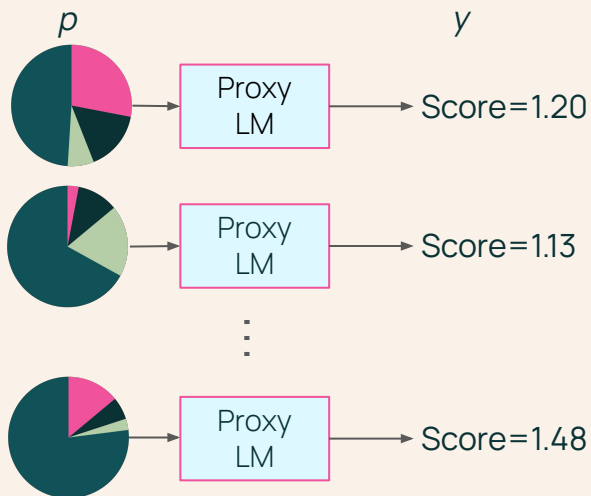
<sup>1</sup>NVIDIA <sup>2</sup>Georgia Institute of Technology

# How data mixing?



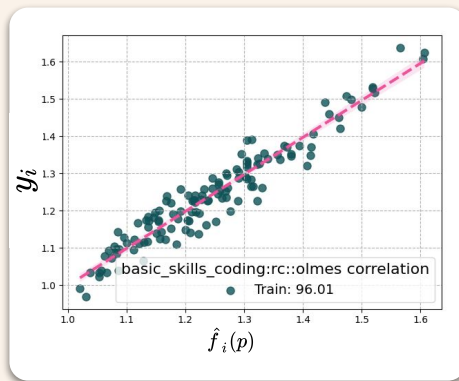
# How data mixing?

1. **Swarm:** Train  $K$  small models with randomly sampled mixtures  $p$



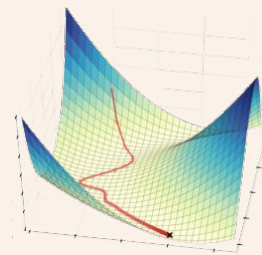
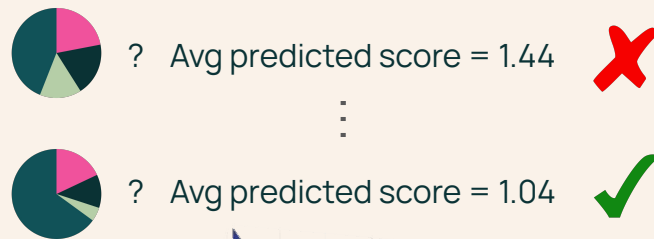
2. **Regression:** Fit function to predict LM performance given mixture  $p$

$$\hat{f}_i(p) \approx y_i$$



3. **Optimize:** Use fit function to solve for optimal mix  $p^*$

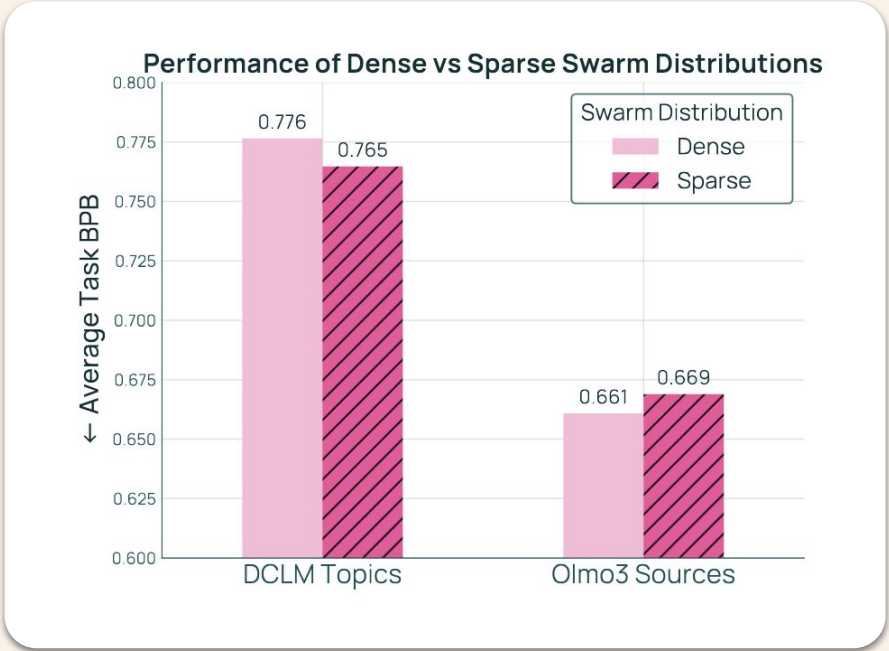
$$\underset{p \in \Delta^{m-1}}{\text{minimize}} \frac{1}{n} \sum_{i=1}^n \hat{f}_i(p)$$



# How data mixing?

Design Choice	RegMix (Liu et al., 2025a)	DML (Ye et al., 2025)	AutoScale (Kang et al., 2025)	BiMix (Ge et al., 2025b)	ADMIRE-BayesOpt (Chen et al., 2025b)	CLIMB (Diao et al., 2025)
<b>Swarm Construction</b>						
Proxy model size	1M	70, 160, 305, 410M	Target	280M	1M, 60M	350M
Swarm size (vs $m$ domains)	512 ( $m = 17$ )	20 ( $m = 7$ )	$2m + 1$	4	101 ( $m = 17$ )	112 ( $m = 21$ )
Swarm distribution	Dirichlet with natural prior	Exponential grid	Exponential grid	Entropy-weighted	Dynamic	Dirichlet with natural prior
<b>Regression Model</b>						
Regression model family	LightGBM	Log-Linear	Power Law	Power Law	Gaussian Process	LightGBM
Regression granularity	Aggregated	Aggregated	Per-Task	Per-Task	Aggregated	Aggregated
<b>Mixture Optimization</b>						
Data repetition constraints	No	No	No	No	No	No
Optimization solver	Search	Search	Gradient Descent	Exact Solver	Search	Search

# Problem 1: No “standard” config



# Problem 1: No “standard” config

## 4.5 Dataset mixing

Often, Common Crawl (CC) is combined with other data sources that are considered high-quality [63, 70, 168, 170] (e.g., Wikipedia, StackExchange, and peS2o [156]). Since DCLM participants can include additional data sources in our mixing track, we examined the potential benefits of adding high-quality sources to training sets derived from Common Crawl only. We compare a model trained on 100% filtered CC data to models trained with the mixing ratios from Llama 1 and RedPajama: 67% CC, and 33% from Wikipedia, Books, Stack exchange, arXiv, and Github. For the CC component, we consider different variants: a subset of our DCLM-BASELINE, RedPajama’s CC portion, RefinedWeb, and C4. The results in Table 6 show that mixing improves performance for the lower-performing CC subsets (C4, RedPajama-CC, and RefinedWeb). In the case of DCLM-BASELINE however, mixing actually hurts performance on average, which suggests it can be counterproductive given performant filtering. For additional mixing results, see Appendix M.

# Problem 2: Data changes during LM development

Olmo 3(2025)

## Add + Remove

- DCLM → Olmo 3 Common Crawl
- Stack v2 → StackEdu
- OpenWebMath → FineMath

## Transform

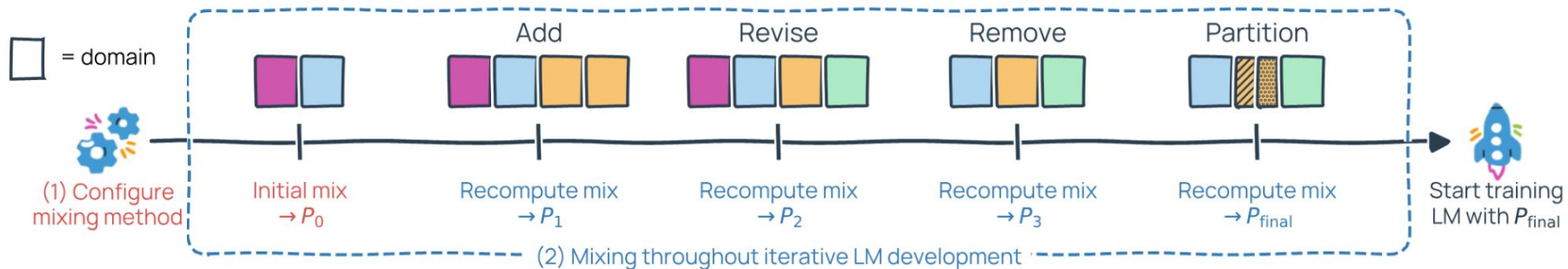
- DCLM quality filter → FineWebEdu quality filter

## Partition

- S2ORC PDFs → Classify by field of study



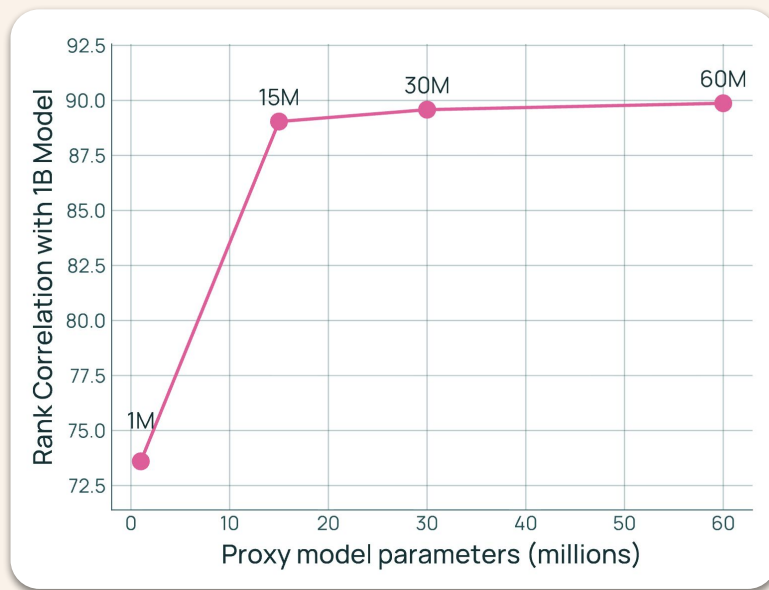
# Problem 2: Data changes during LM development



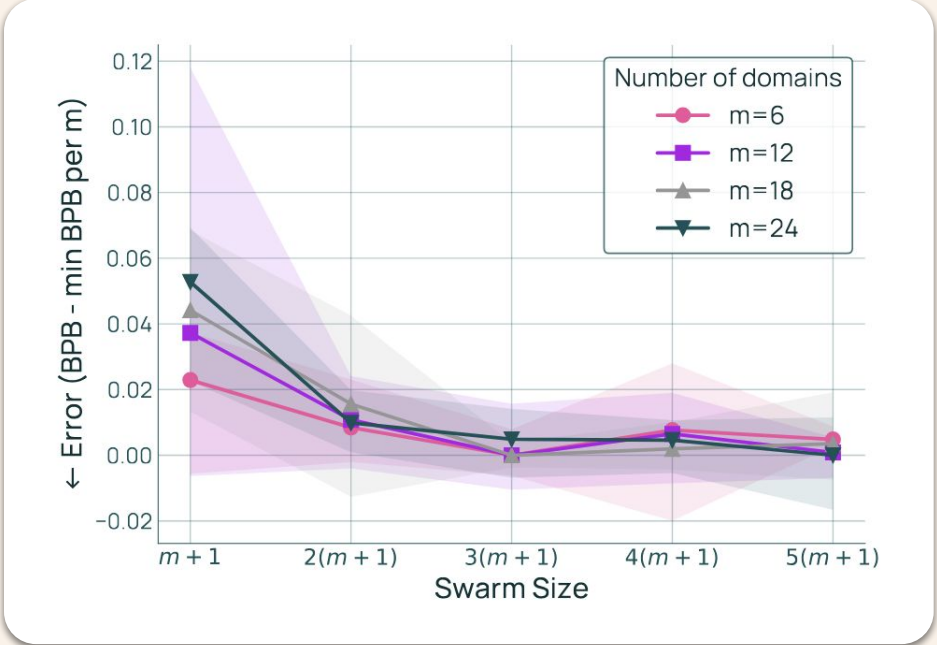
# Olmix: Data mixing over the language model development cycle

# Finding 1: Find smallest model size where performance ranking correlates well with larger models

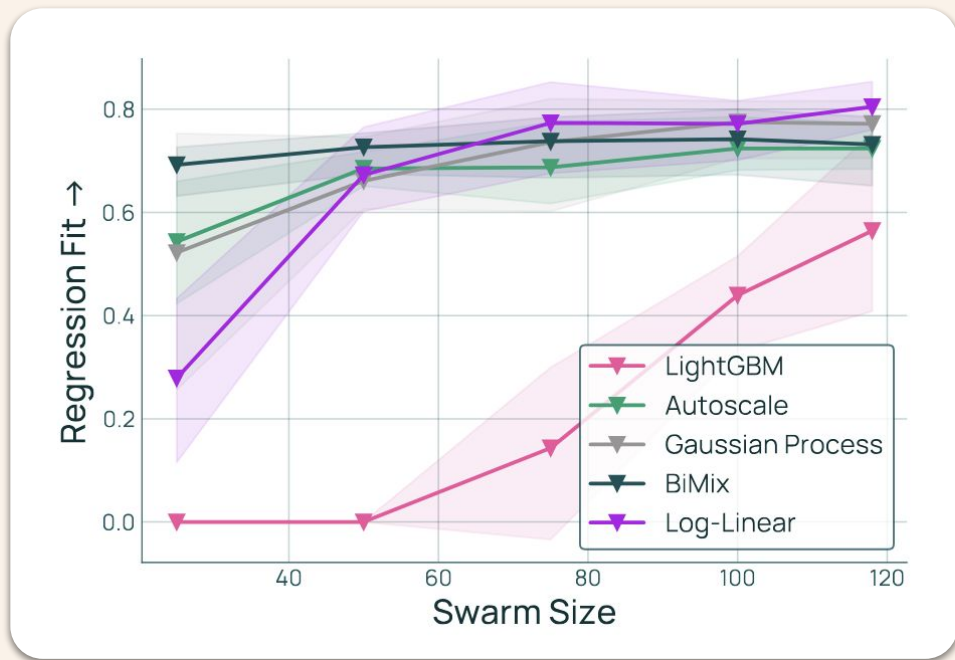
Regmix paper →



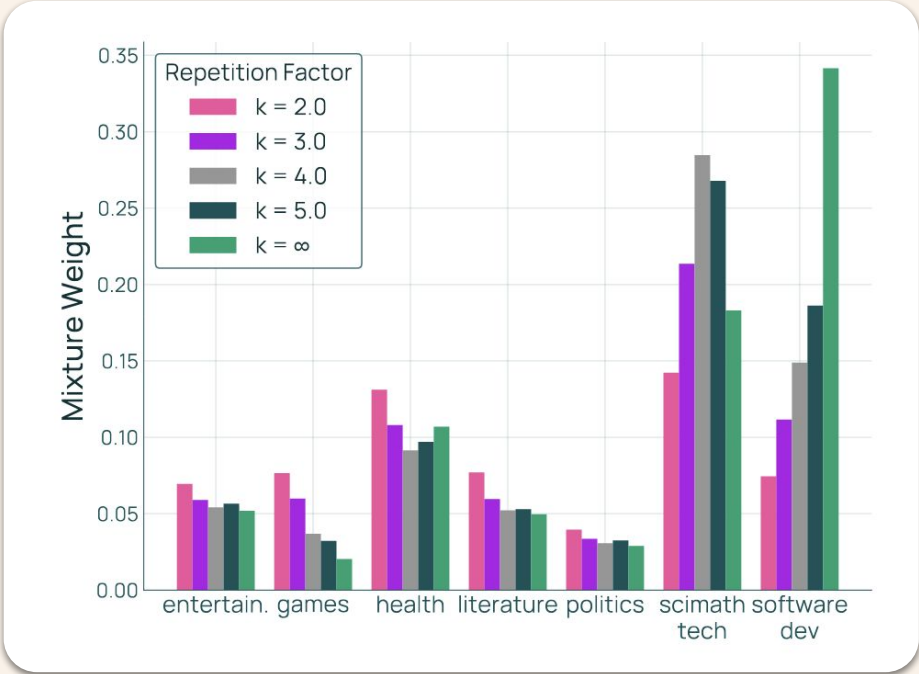
# Finding 2: Swarm size scales linearly with number of domains



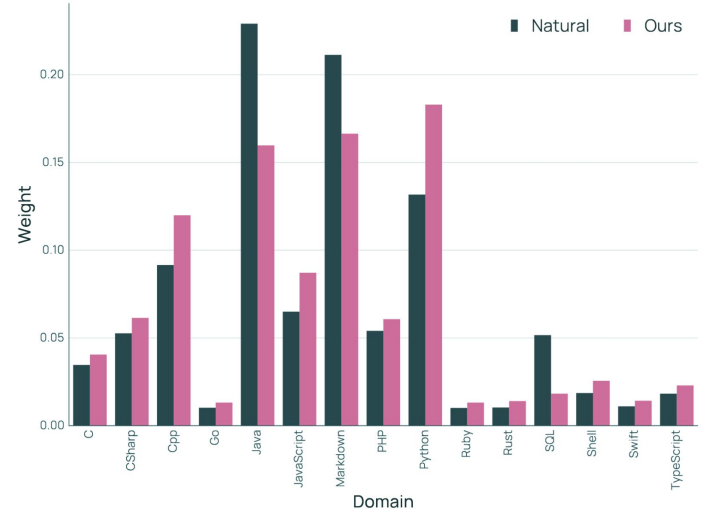
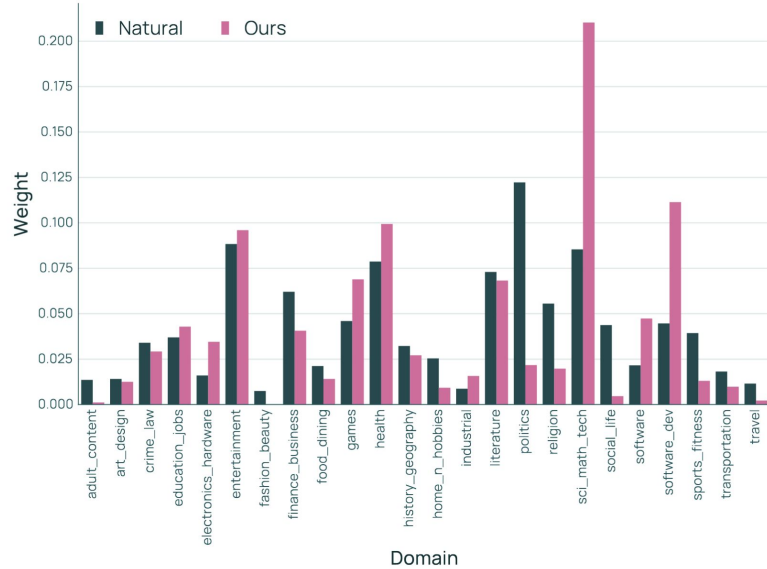
## Finding 3: Given sufficient compute, regression functional forms all similar



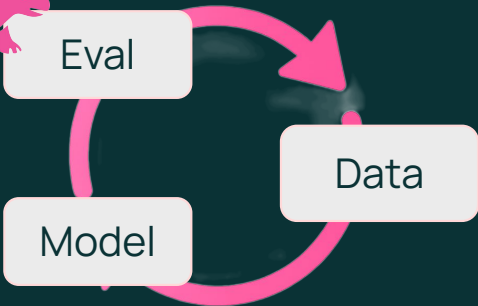
# Finding 4: Data mixing dramatically sensitive if forced to repeat data



# Now we can mix data!



We are here!



## Olmo 3

Olmo Team\*

Allyson Ettlinger<sup>\*1</sup> Amanda Bertsch<sup>\*1,3</sup> Bailey Kuehl<sup>\*1</sup> David Graham<sup>\*1</sup>  
 David Heineman<sup>\*1</sup> Dirk Groeneveld<sup>\*1</sup> Faeze Brahman<sup>\*1</sup> Finbarr Timbers<sup>\*1</sup>  
 Hamish Ivison<sup>\*1,2</sup> Jacob Morrison<sup>\*1,2</sup> Jake Poznanski<sup>\*1</sup> Kyle Lo<sup>\*1,2</sup> Luca Soldaini<sup>\*1</sup>  
 Matt Jordan<sup>\*1</sup> Mayee Chen<sup>\*1,4</sup> Michael Noukhovitch<sup>\*1,5,6</sup> Nathan Lambert<sup>\*1</sup>  
 Pete Walsh<sup>\*1</sup> Pradeep Dasigi<sup>\*1</sup> Robert Berry<sup>\*1</sup> Saumya Malik<sup>\*1</sup> Saurabh Shah<sup>\*1</sup>  
 Scott Geng<sup>\*1,2</sup> Shane Arora<sup>\*1</sup> Shashank Gupta<sup>\*1</sup> Taira Anderson<sup>\*1</sup> Teng Xiao<sup>\*1</sup>  
 Tyler Murray<sup>\*1</sup> Tyler Romero<sup>\*1</sup> Victoria Graf<sup>\*1,2</sup>

Akari Asai<sup>1,3</sup> Akshita Bhagia<sup>1</sup> Alexander Wettig<sup>1</sup> Alisa Liu<sup>2</sup> Aman Rangapur<sup>1</sup>  
 Chloe Anastasiades<sup>1</sup> Costa Huang<sup>1</sup> Dustin Schwenk<sup>1</sup> Harsh Trivedi<sup>1</sup> Ian Magnusson<sup>1,2</sup>  
 Jaron Lochner<sup>1</sup> Jiacheng Liu<sup>1</sup> Lester James V. Miranda<sup>1</sup> Maarten Sap<sup>1,3</sup> Malia Morgan<sup>1</sup>  
 Michael Schmitz<sup>1</sup> Michal Querquin<sup>1</sup> Michael Wilson<sup>1</sup> Regan Huff<sup>1</sup> Ronan Le Bras<sup>1</sup>  
 Rui Xin<sup>2</sup> Rulin Shao<sup>2</sup> Sam Sijonasberg<sup>1</sup> Shannon Zhai<sup>1</sup> Shen<sup>1</sup> Shuyue Stella Li<sup>1</sup>  
 Tucker Wilde<sup>1</sup> Valentina Pyatkin<sup>1</sup> Will Merrill<sup>1</sup> Yapei Chang<sup>2</sup> Yuling Gu<sup>1</sup> Zhiyuan Zeng<sup>1,2</sup>

Ashish Sabharwal<sup>1</sup> Luke Zettlemoyer<sup>2</sup> Pang Wei Koh<sup>1,2</sup>  
 Ali Farhadi<sup>1,2</sup> Noah A. Smith<sup>\*1,2</sup> Hannaneh Hajishirzi<sup>\*1,2</sup>

<sup>1</sup>Allen Institute for AI <sup>2</sup>University of Washington <sup>3</sup>Carnegie Mellon University <sup>4</sup>Stanford University <sup>5</sup>Mila  
<sup>6</sup>Université de Montréal <sup>7</sup>Princeton University <sup>8</sup>Massachusetts Institute of Technology <sup>9</sup>University of Maryland

\*OLMO 3 was a team effort; authors sorted alphabetically. \*marks core contributors. See author contributions here.

- 🟡 **Olmo 3 Base:** Olmo-3-1025-7B Olmo-3-1125-32B
- 🟡 **Olmo 3 Think:** Olmo-3-7B-Think Olmo-3-13-32B-Think
- 🟡 **Olmo 3 Instruct:** Olmo-3-7B-Instruct Olmo-3-1-32B-Instruct
- 🟡 **Olmo 3 RL Zero:** Olmo-3-7B-RL-Zero-(MathCode|IF|General|Mix) Olmo-3-1-7B-RL-Zero-(Math|Code)
- 🟢 **Base Data:** Pretrain: Dolma 3 Mix Midtrain: Dolma 3 Dolmino Mix Long-ctx: Dolma 3 Longino Mix
- 🟢 **Think Data:** Dolci-Think-(SFT|DPO|RL)-7B Dolci-Think-(SFT|DPO|RL)-32B
- 🟢 **Instruct Data:** Dolci-Instruct-(SFT|DPO|RL)
- 🟢 **RL-Zero Data:** Dolci-RL-Zero-(Math|Code|IF|General)-7B Dolci-RL-Zero-Mix-7B

### 3.3 Experimental Design and Evaluation

Model development requires many iterative data and training decisions. However, benchmarks are not perfect decision-making tools: different evaluations are only sensitive for making development decisions across specific ranges of scale and capability (Magnusson et al., 2025). Models trained at small compute scales are known to exhibit random-chance performance on math, code, and multiple-choice question answering (MCQA) tasks (Wei et al., 2022; Gu et al., 2024b), and benchmark noise can reduce the ability to trust small differences in scores (Heineman et al., 2025). To address these problems, we develop **OLMOBASEEVAL**, a collection of benchmark suites to support decision-making during base model development. **OLMOBASEEVAL** features the

We release **OLMO 3**, a family of state-of-the-art, fully open language models at the 7B and 32B parameter scales. **OLMO 3** model construction targets long-context reasoning, function calling, coding, instruction following, general chat, and knowledge recall. This release includes the entire model flow, i.e., the full lifecycle of the family of models, including every stage, checkpoint, data point, and dependency used to build it. Our flagship model, **OLMO 3.1 THINK 32B**, is the strongest fully-open thinking model released to-date.

# OLMo 2 → Olmo 3 base evals:

- + capabilities (math, code)
- + coverage of science QA, Basic Skills
- + coverage across formats (Gen2MC)

## 3 new methods for Olmo 3 Base eval:

- (1) clustering tasks,
- (2) scaling analysis,
- (3) signal-to-noise analysis

task	split	# inst (total)	# shots	metric	reference
<b>Multiple-choice tasks</b>					
ARC-Challenge (ARC_C)	Test	1172	5	pmi	(Clark et al., 2018)
BoolQ	Val	1000 (3270)	5	none	(Clark et al., 2019)
HellaSwag (HSwag)	Val	1000 (10042)	5	char	(Zellers et al., 2019)
MMLU <sup>†</sup>	Test	14042	5	char	(Hendrycks et al., 2021a)
WinoGrande (WinoG)	Val	1267	5	none	(Sakaguchi et al., 2020)
<b>Generative tasks</b>					
DROP	Val	1000 (9536)	5	F1	(Dua et al., 2019)
Natural Questions (NatQs)	Val	1000 (3610)	5	F1	(Kwiatkowski et al., 2019)
<b>Held-out tasks</b>					
AGIEval English	Test	2646	1	MCF	(Zhong et al., 2024)
GSM8K	Test	1319	8 (CoT)	EM	(Cobbe et al., 2021)
MMLU-Pro	Test	12032	5	MCF	(Wang et al., 2024)
TriviaQA	Val	7993	5	F1	(Joshi et al., 2017)

**Table 20** Details of OLMES benchmarks used in OLMo 2 evaluation, with standardized choices of dataset split, number of instances to use, along with total number if sampling was used. For multiple-choice tasks, when using the Cloze/Completion Formulation (CF), the “metric” column specifies which normalization scheme to use. Following the OLMES standard, we evaluate each model using both the MCF (Multiple-Choice Formulation) and CF formulations, and the best performing one is used. For efficiency reasons, we limit MMLU and held-out multiple-choice evaluations to MCF only as all the relevant models strongly prefer that format for these tasks.

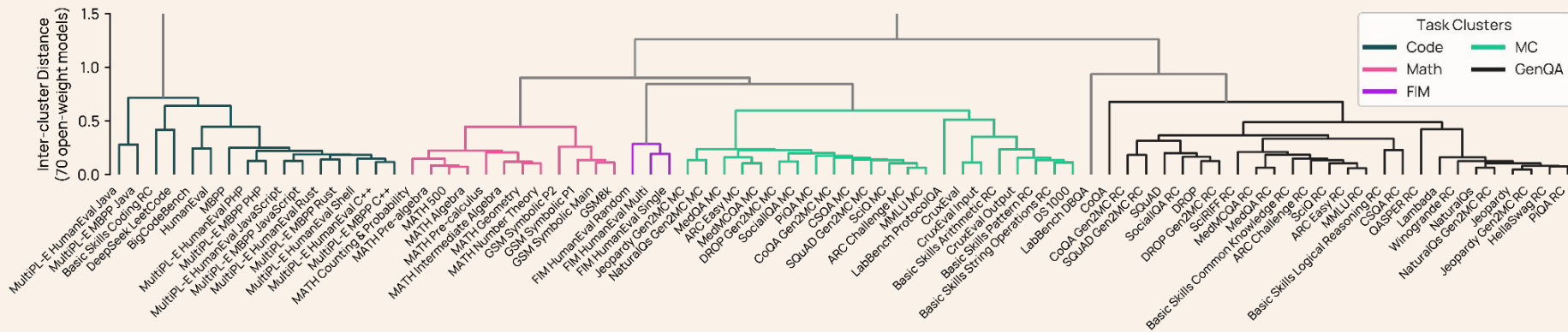


task	ICL	format	metric	temp	top-p	max toks	p@k (n)	# sub	
<b>Base Main Suite</b>									
Math	GSM8K* (2021)	8 <sup>α</sup>	CoT EM	pass@k	0.6	0.6	512	1, 4 (8)	-
	GSM Symbolic* (2024)	8 <sup>α</sup>	CoT EM	pass@k	0.6	0.6	512	1, 4 (8)	3
	Minerva MATH* (2022)	4 <sup>α</sup>	CoT EM	pass@k	0.6	0.6	1024	1, 4 (4)	7
	MATH 500* (2022; 2023)	4 <sup>α</sup>	CoT EM	pass@k	0.6	0.6	1024	1, 16 (32)	-
Code	HumanEval* (2021)	3	Code Exec	pass@k	0.6	0.6	512	1, 16 (32)	-
	MBPP* (2021)	3	Code Exec	pass@k	0.6	0.6	512	1, 16 (32)	-
	BigCodeBench* (2024)	3	Code Exec	pass@k	0.6	0.6	1280	1 (5)	-
	DS 1000* (2022)	3	Code Exec	pass@k	0.6	0.6	1024	1 (5)	-
	Deepseek LeetCode* (2024)	0	Code Exec	pass@k	0.6	0.6	512	1, 16 (32)	-
	MultiPL-E HumanEval* (2022)	0	Code Exec	pass@k	0.6	0.6	1024	1, 16 (32)	6
	MultiPL-E MBPP* (2022)	0	Code Exec	pass@k	0.6	0.6	1024	1, 16 (32)	6
FIM	HumEval FIM Single* (2022)	0	FIM	pass@1	0.8	0.95	512	1 (10)	-
	HumEval FIM Random* (2022)	0	FIM	pass@1	0.8	0.95	512	1 (5)	-
	HumEval FIM Multi* (2022)	0	FIM	pass@1	0.8	0.95	512	1 (1)	-
STEM QA	ARC (2018)	5	MC	Acc	-	-	-	-	2
	MMLU STEM (2021b)	5	MC	Acc	-	-	-	-	19
	MedMCQA* (2022)	5	MC	Acc	-	-	-	-	-
	MedQA* (2021)	5	MC	Acc	-	-	-	-	-
	SciQ* (2017)	5	MC	Acc	-	-	-	-	-
	MMLU Humanities (2021b)	5	MC	Acc	-	-	-	-	13
	MMLU Social Sci. (2021b)	5	MC	Acc	-	-	-	-	12
Non-STEM QA	MMLU Other (2021b)	5	MC	Acc	-	-	-	-	14
	CSQA (2019)	5	MC	Acc	-	-	-	-	-
	PIQA (2020)	5	MC	Acc	-	-	-	-	-
	SocialIQA (2019)	5	MC	Acc	-	-	-	-	-
	DROP Gen2MC* (§A.3.2; 2019)	5	MC	Acc	-	-	-	-	-
	Jeopardy Gen2MC* (§A.3.2; 2024)	5	MC	Acc	-	-	-	-	-
	NaturalQs Gen2MC* (§A.3.2; 2019)	5	MC	Acc	-	-	-	-	-
	SQuAD Gen2MC* (§A.3.2; 2016)	5	MC	Acc	-	-	-	-	-
	CoQA Gen2MC* (§A.3.2; 2019)	0 <sup>†</sup>	MC	Acc	-	-	-	-	-
	Basic Skills* (§A.3.2)	5	MC	Acc	-	-	-	-	6
GenQA	HellaSwag (2019)	5	RC <sub>per-char</sub>	Acc	-	-	-	-	-
	WinoGrande (2020)	5	RC <sub>none</sub>	Acc	-	-	-	-	-
	Lambda (2016)	0	RC <sub>per-char</sub>	Acc	-	-	-	-	-
	Basic Skills* (§A.3.2)	5	RC <sub>per-token</sub>	Acc	-	-	-	-	6
	DROP (2019)	5	GenQA	F1	0	1	100	-	-
	Jeopardy (2024)	5	GenQA	F1	0	1	50	-	-
	NaturalQs (2019)	5	GenQA	F1	0	1	50	-	-
	SQuAD (2016)	5	GenQA	F1	0	1	50	-	-
	CoQA (2019)	0 <sup>†</sup>	GenQA	F1	0	1	50	-	-
<b>Base Held-out Suite</b>									
MMLU Pro (2024a)	5	MC	Acc	-	-	-	-	13	
LBPP* (2024)	0	Code Exec	pass@k	0.6	0.6	4096	1 (32)	-	
Deepmind Math* (2019)	5	CoT EM	pass@k	0.6	0.6	2048	1 (1)	-	
BigBench Hard (2022)	3	CoT EM	Acc	0.6	0.6	512	1 (1)	55	

**Table 43** Details of the OLMo 3 base evaluation suite. Tasks were formatted as multiple-choice (MC), rank choice (RC), following the setup in Gu et al. (2024b), short-form generative (GenQA), chain-of-thought with exact-match scoring (CoT EM), code execution (Code Exec) or fill-in-the-middle coding (FIM). \* = new additions to the base OLMo 2 suite (OLMo et al., 2024); <sup>†</sup> = few-shot examples are built-in the task; <sup>α</sup> = human-written few-shot examples.

1

# How to handle large # of benchmarks? Group tasks into “clusters” (Math, Code, GenQA) and track multi-task averages



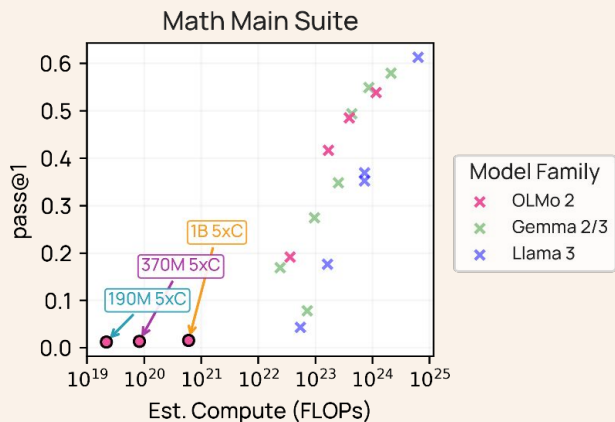
Task averages simplify decision making (e.g. in midtraining)

The image displays a large spreadsheet with multiple columns and rows, representing performance metrics for various models and tasks. A red arrow points from the text to a specific row in the spreadsheet. A blue box highlights a summary table for 'OLMOBASEVAL'.

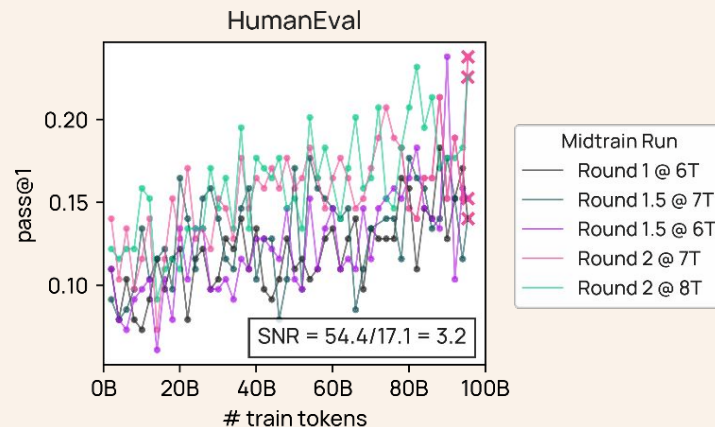
Mix	OLMOBASEVAL							SFT Exps	
	Avg	MC STEM	MC Non-STEM	GenQA	Math	Code	FIM	Avg	
Round 1	49.7	64.3	75.2	68.3	47.4	23.4	28.4	35.2	
Round 3	50.7	64.9	75.7	68.1	48.7	24.4	31.9	35.3	
Round 5	53.1	65.3	76.1	70.8	57.1	27.7	29.4	37.3	

2 + 3

## Two additional problems:

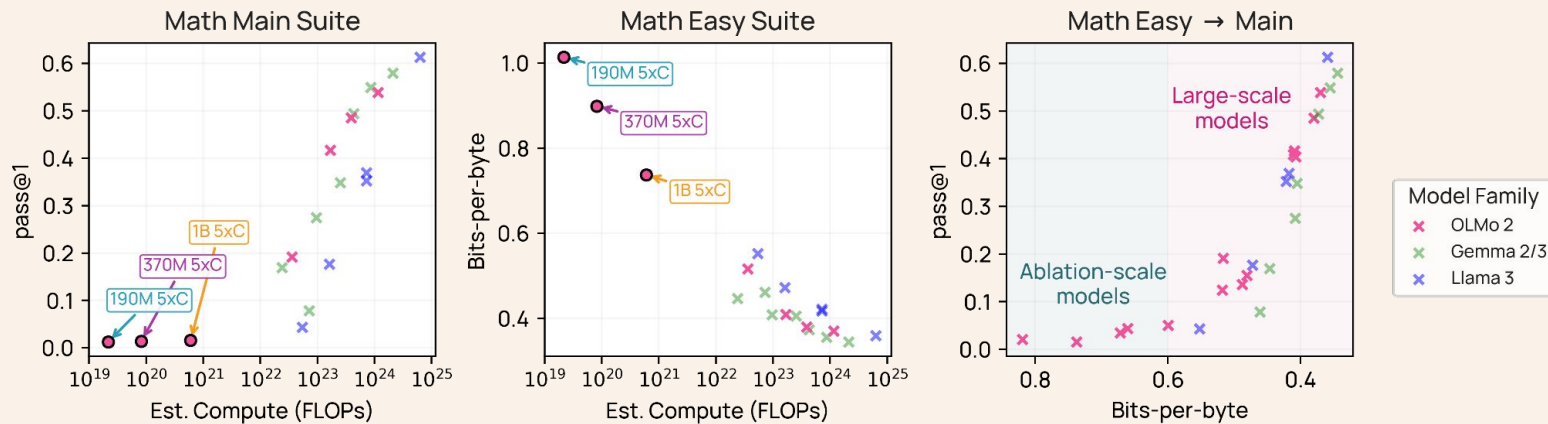


**Signal at small scales**  
(e.g. 30M to 1B)



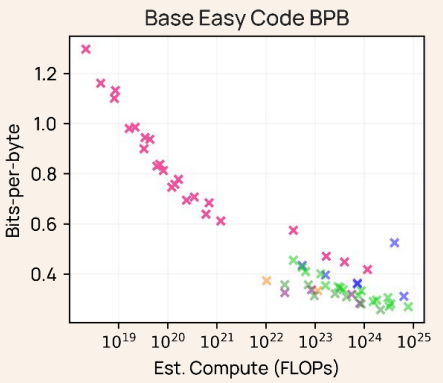
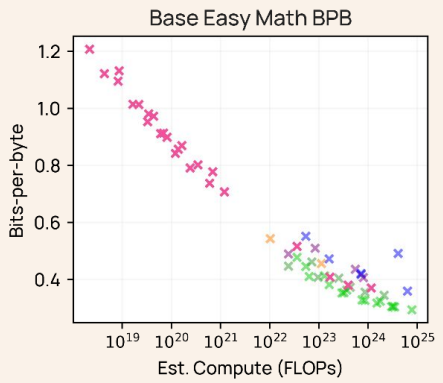
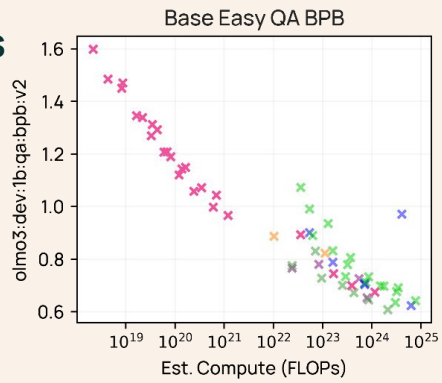
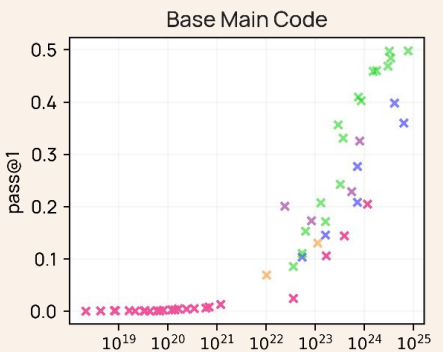
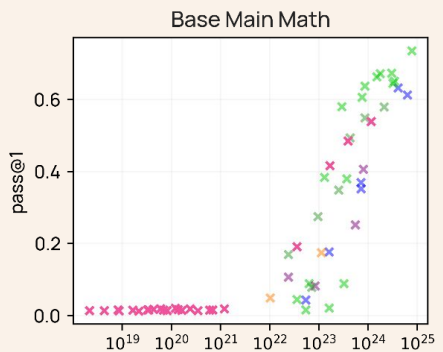
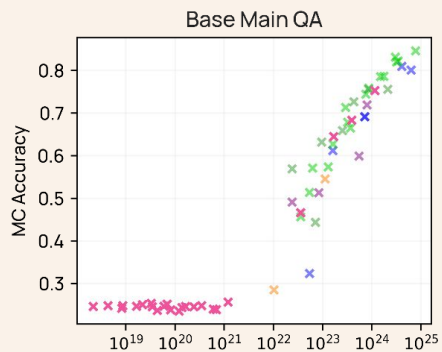
**Noise at large scales**  
(e.g. midtraining)

## ② Small-scale metrics:



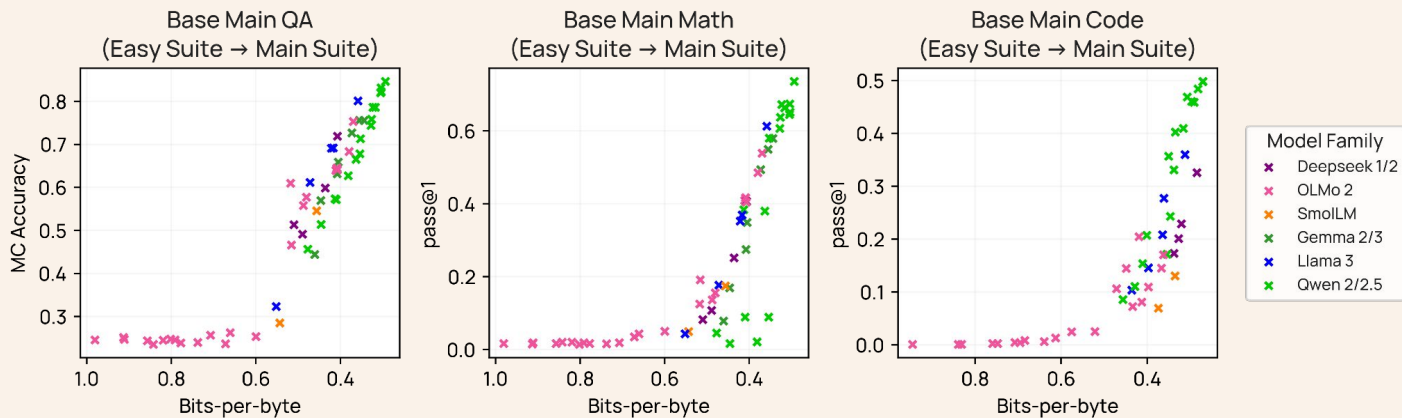
## ② Small-scale metrics:

Separate decision-making suite using BPB over continuations

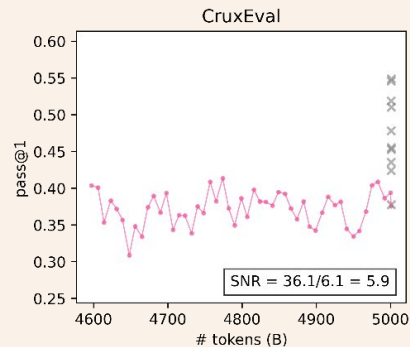
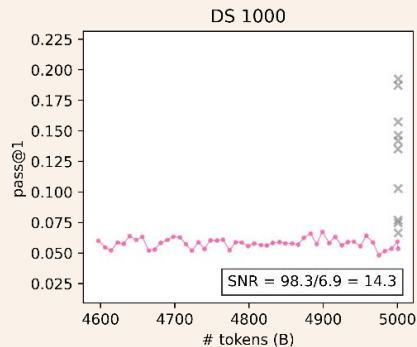
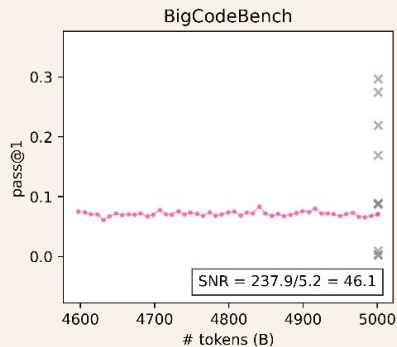
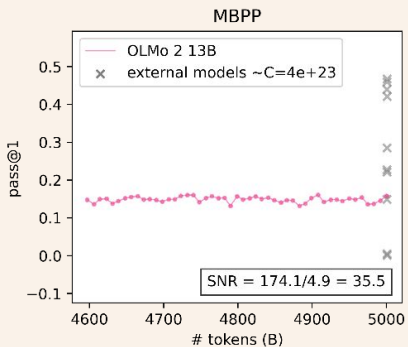
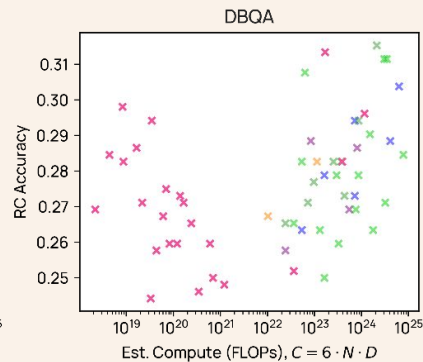
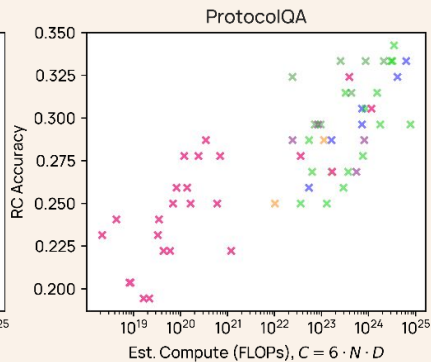
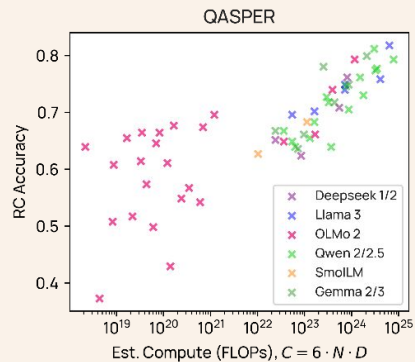
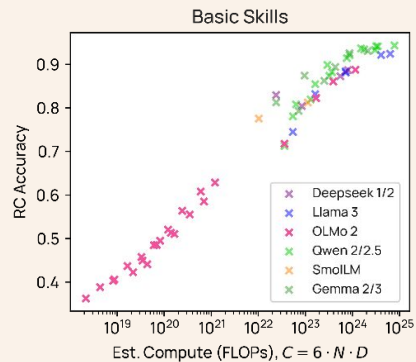


- Model Family
- Deepseek 1/2
  - Llama 3
  - OLMo 2
  - Qwen 2/2.5
  - SmoLM
  - Gemma 2/3

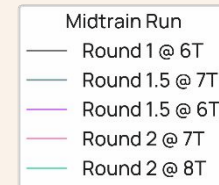
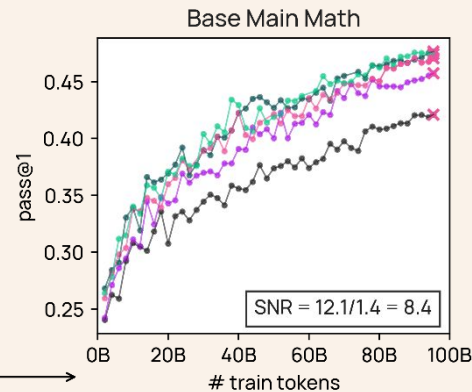
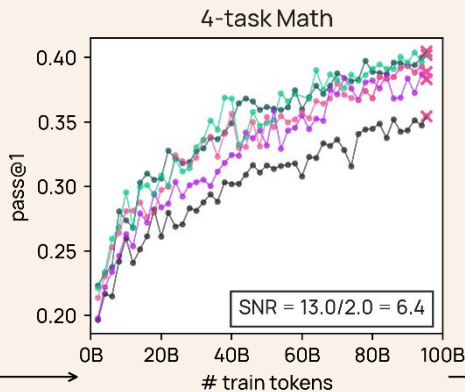
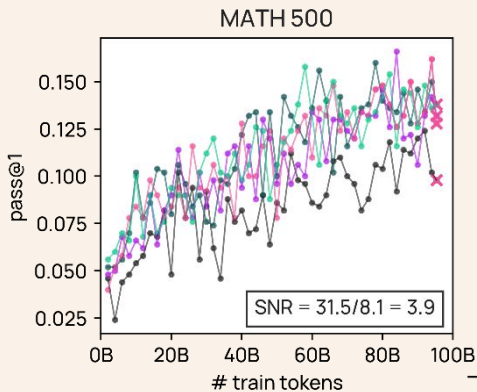
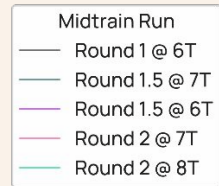
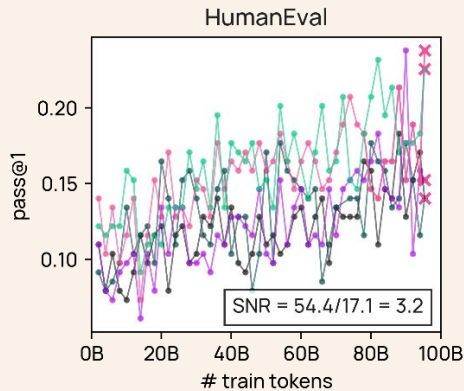
## ② Small-scale metrics:

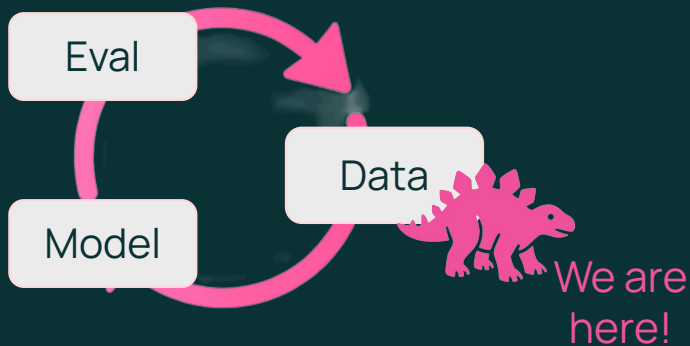


# 3 SNR in Olmo 3 Eval



# 3 SNR in Olmo 3 Eval





## Olmo 3

Olmo Team\*

Allyson Ettlinger<sup>\*1</sup> Amanda Bertsch<sup>\*1,3</sup> Bailey Kuehl<sup>\*1</sup> David Graham<sup>\*1</sup>  
 David Heineman<sup>\*1</sup> Dirk Groeneveld<sup>\*1</sup> Faeze Brahman<sup>\*1</sup> Finbarr Timbers<sup>\*1</sup>  
 Hamish Ivison<sup>\*1,2</sup> Jacob Morrison<sup>\*1,2</sup> Jake Poznanski<sup>\*1</sup> Kyle Lo<sup>\*1,2</sup> Luca Soldaini<sup>\*1</sup>  
 Matt Jordan<sup>\*1</sup> Mayee Chen<sup>\*1,4</sup> Michael Noukhovitch<sup>\*1,5,6</sup> Nathan Lambert<sup>\*1</sup>  
 Pete Walsh<sup>\*1</sup> Pradeep Dasigi<sup>\*1</sup> Robert Berry<sup>\*1</sup> Saumya Malik<sup>\*1</sup> Saurabh Shah<sup>\*1</sup>  
 Scott Geng<sup>\*1,2</sup> Shane Arora<sup>\*1</sup> Shashank Gupta<sup>\*1</sup> Taira Anderson<sup>\*1</sup> Teng Xiao<sup>\*1</sup>  
 Tyler Murray<sup>\*1</sup> Tyler Romero<sup>\*1</sup> Victoria Graf<sup>\*1,2</sup>

Akari Asai<sup>1,3</sup> Akshita Bhagia<sup>1</sup> Alexander Wettig<sup>1</sup> Alisa Liu<sup>2</sup> Aman Rangapur<sup>1</sup>  
 Chloe Anastasiades<sup>1</sup> Costa Huang<sup>1</sup> Dustin Schwenk<sup>1</sup> Harsh Trivedi<sup>1</sup> Ian Magnusson<sup>1,2</sup>  
 Jaron Lochner<sup>1</sup> Jiacheng Liu<sup>1</sup> Lester James V. Miranda<sup>1</sup> Maarten Sap<sup>1,3</sup> Malia Morgan<sup>1</sup>  
 Michael Schmitz<sup>1</sup> Michal Querquin<sup>1</sup> Michael Wilson<sup>1</sup> Regan Huff<sup>1</sup> Ronan Le Bras<sup>1</sup>  
 Rui Xin<sup>2</sup> Rulin Shao<sup>2</sup> Sam Sijmonsberg<sup>2</sup> Shannon Zaijiang Shen<sup>1</sup> Shuyue Stella LF  
 Tucker Wilde<sup>1</sup> Valentina Pyatkin<sup>1</sup> Will Merrill<sup>1</sup> Yapei Chang<sup>2</sup> Yuling Gu<sup>1</sup> Zhiyuan Zeng<sup>1,2</sup>

Ashish Sabharwal<sup>1</sup> Luke Zettlemoyer<sup>2</sup> Pang Wei Koh<sup>1,2</sup>  
 Ali Farhadi<sup>1,2</sup> Noah A. Smith<sup>\*1,2</sup> Hannaneh Hajishirzi<sup>\*1,2</sup>

<sup>1</sup>Allen Institute for AI <sup>2</sup>University of Washington <sup>3</sup>Carnegie Mellon University <sup>4</sup>Stanford University <sup>5</sup>Mila  
<sup>6</sup>Université de Montréal <sup>7</sup>Princeton University <sup>8</sup>Massachusetts Institute of Technology <sup>9</sup>University of Maryland

\*OLMO 3 was a team effort; authors sorted alphabetically. \*marks core contributors. See author contributions here.

- 🟡 **Olmo 3 Base:** [Olmo-3-1025-7B](#) [Olmo-3-1125-32B](#)
- 🟡 **Olmo 3 Think:** [Olmo-3-7B-Think](#) [Olmo-3\(12-1\)-32B-Think](#)
- 🟡 **Olmo 3 Instruct:** [Olmo-3-7B-Instruct](#) [Olmo-3-1-32B-Instruct](#)
- 🟡 **Olmo 3 RLZero:** [Olmo-3-7B-RL-Zero-\(Math|Code|IF|General|Mix\)](#) [Olmo-3-1-7B-RL-Zero-\(Math|Code\)](#)
- 🟢 **Base Data:** [Pretrain: Dolma 3 Mix](#) [Midtrain: Dolma 3 Dolmino Mix](#) [Long-ctx: Dolma 3 Longino Mix](#)
- 🟢 **Think Data:** [Dolci-Think-\(SFT|DP0|RL\)-7B](#) [Dolci-Think-\(SFT|DP0|RL\)-32B](#)

### 3.4 Stage 1: Pretraining

We first train **OLMO 3 BASE** on **DOLMA 3 MIX**, our 6T token pretraining data mix. While **DOLMA 3 MIX** is comprised of largely the same types of data sources used in other open pretraining recipes ([Soldaini et al., 2024](#); [Bakoch et al., 2025](#); [OLMo et al., 2024](#)), we demonstrate three key novelties:

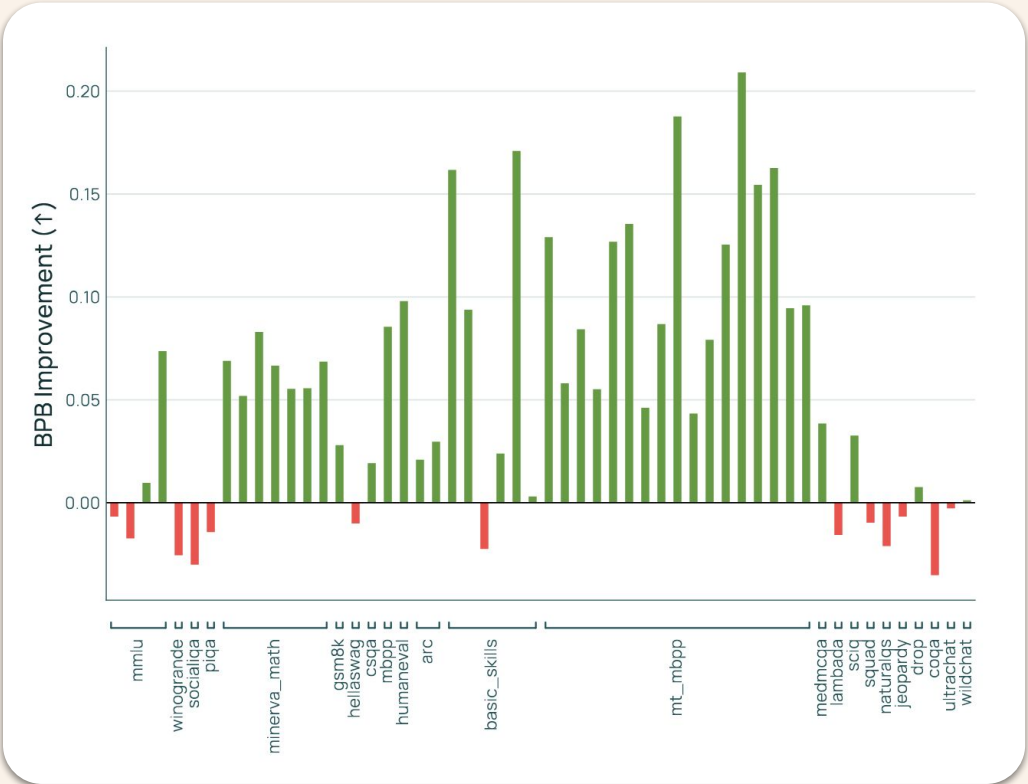
📍 [Contact: olmo@allenai.org](mailto:olmo@allenai.org)

### Abstract

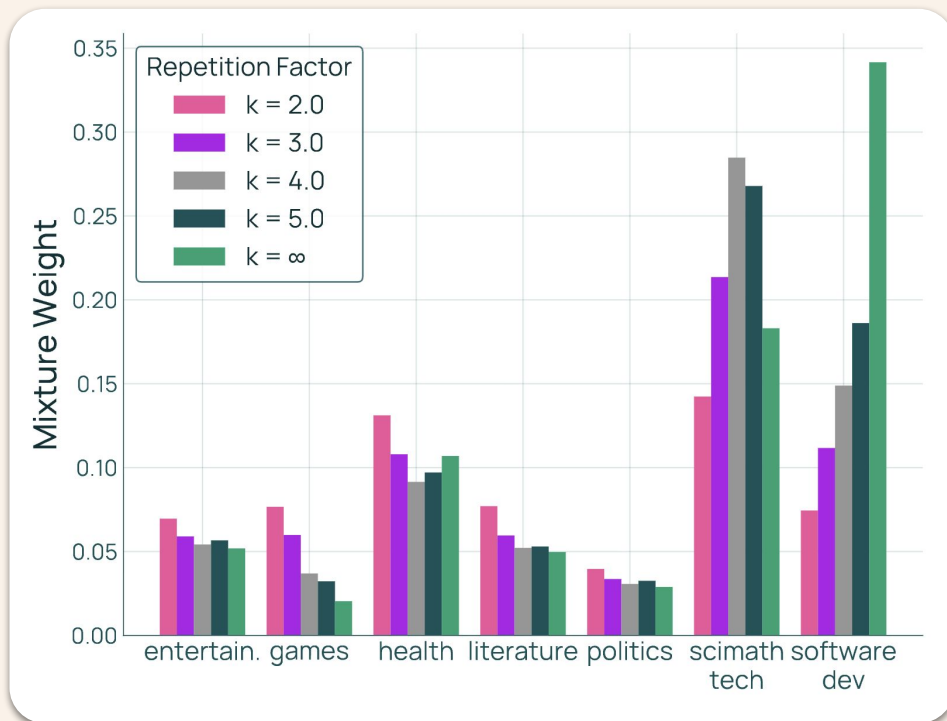
🔗 [Ai2](#)

We introduce **OLMO 3**, a family of state-of-the-art, fully-open language models at the 7B and 32B parameter scales. **OLMO 3** model construction targets long-context reasoning, function calling, coding, instruction following, general chat, and knowledge recall. This release includes the entire **model flow**, i.e., the full lifecycle of the family of models, including every stage, checkpoint, data point, and dependency used to build it. Our flagship model, **OLMO 3.1 THINK 32B**, is the strongest fully-open thinking model released to-date.

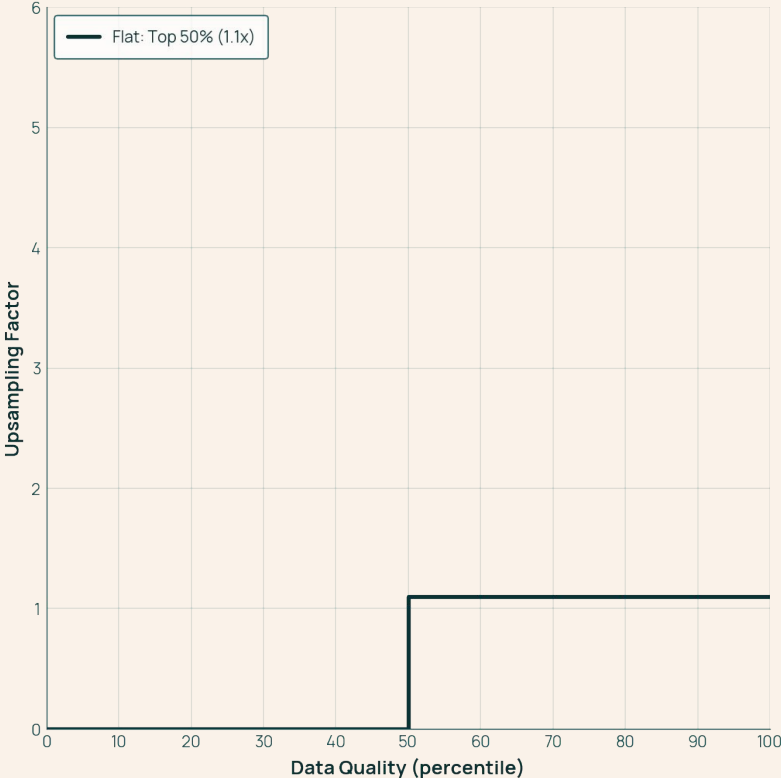
# Data mixing: Operations



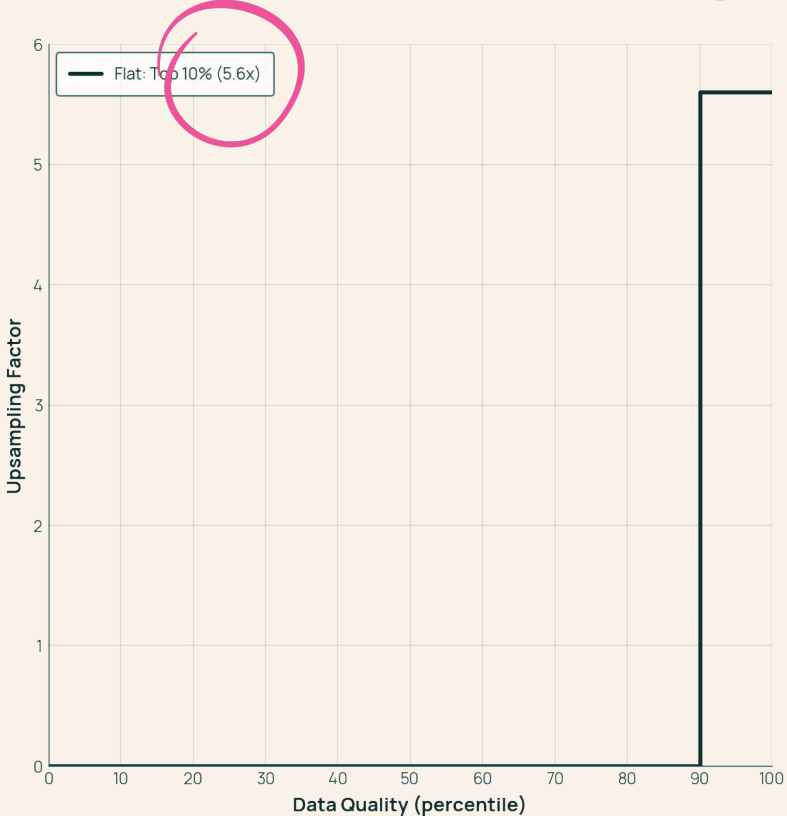
# Data mixing under data constraints



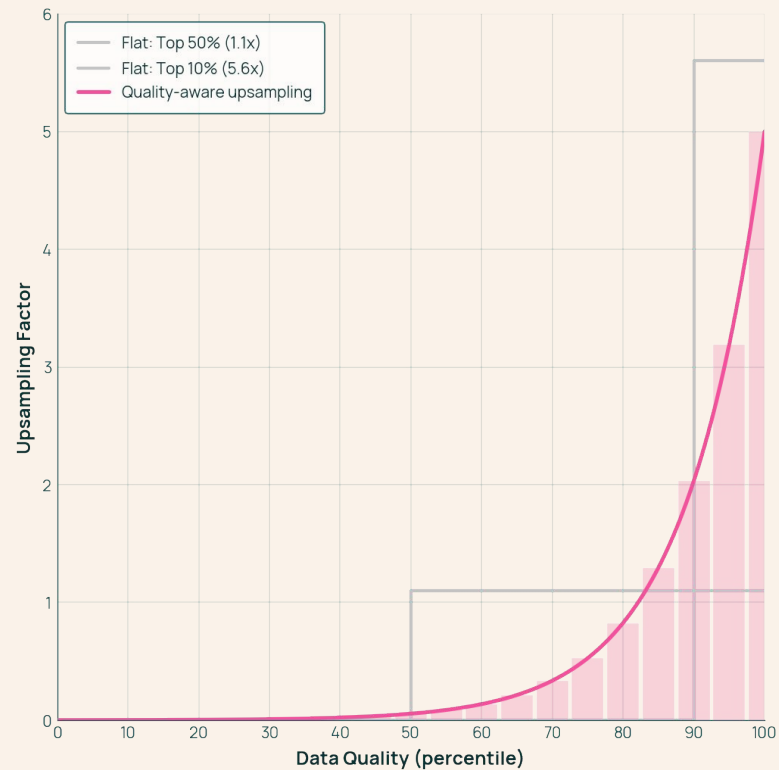
# Data constraints influence upsampling



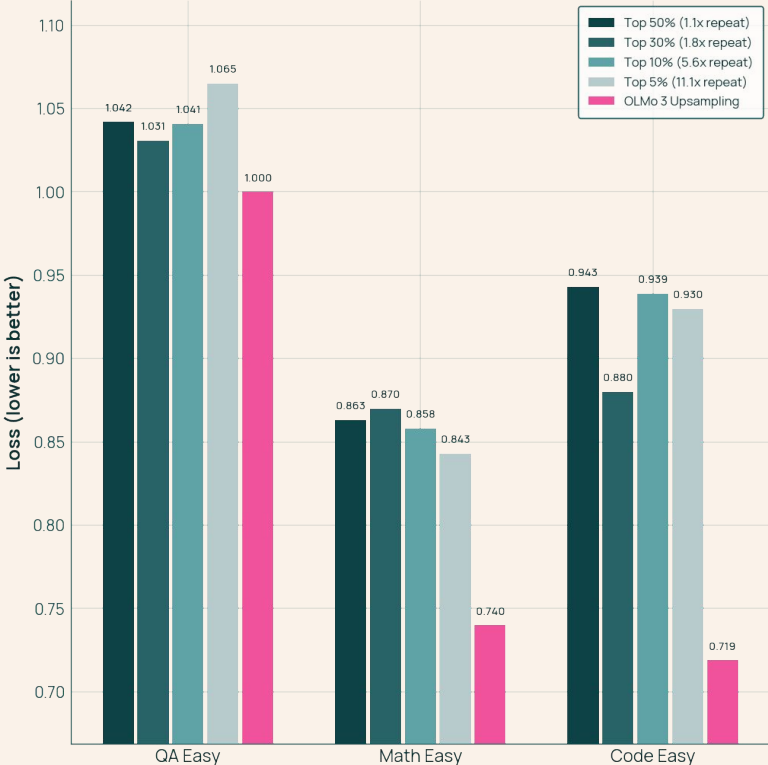
# Data constraints influence upsampling



# Quality-aware upsampling



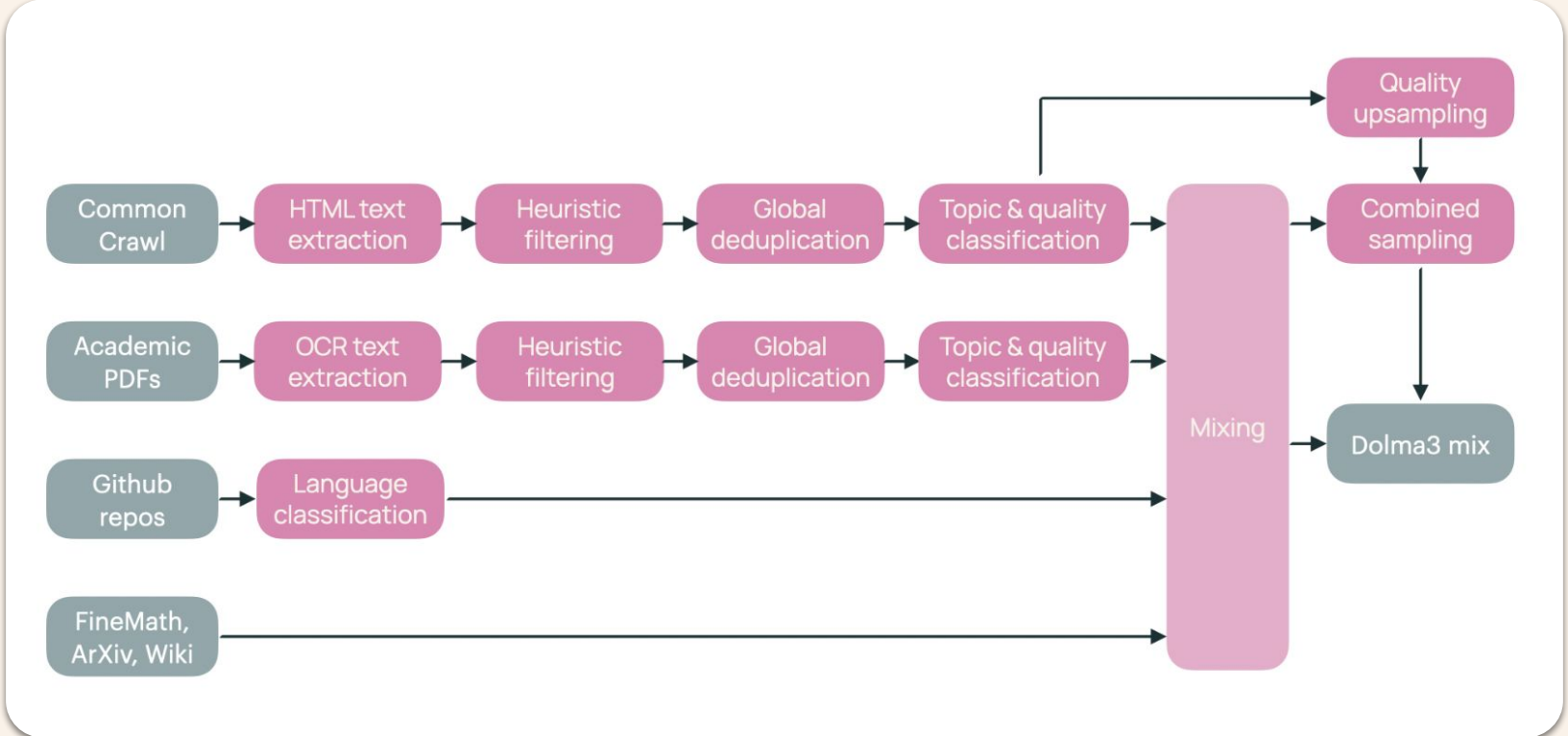
# Quality aware upsampling



# Pretraining: Dolma 3

Source	Type	9T Pool		6T Mix	
		Tokens	Docs	Tokens	Docs
Common Crawl	Web pages	8.14T	9.67B	4.51T (76.1%)	3.15B
OLMOOCR science PDFs	Academic documents	972B	101M	805B (13.6%)	83.8M
Stack-Edu (Rebalanced)	GitHub code	137B	167M	409B (6.89%)	526M
arXiv	Papers with LaTeX	21.4B	3.95M	50.8B (0.86%)	9.10M
FineMath 3+	Math web pages	34.1B	21.4M	152B (2.56%)	95.5M
Wikipedia & Wikibooks	Encyclopedic	3.69B	6.67M	2.51B (0.04%)	4.24M
<b>Total</b>		<b>9.31T</b>	<b>9.97B</b>	<b>5.93T (100%)</b>	<b>3.87B</b>

Extraction, filtering, deduplication, quality classification



# Filtering, Deduplication, Fine-Grained Organization

Common Crawl



255B Docs



39B Docs

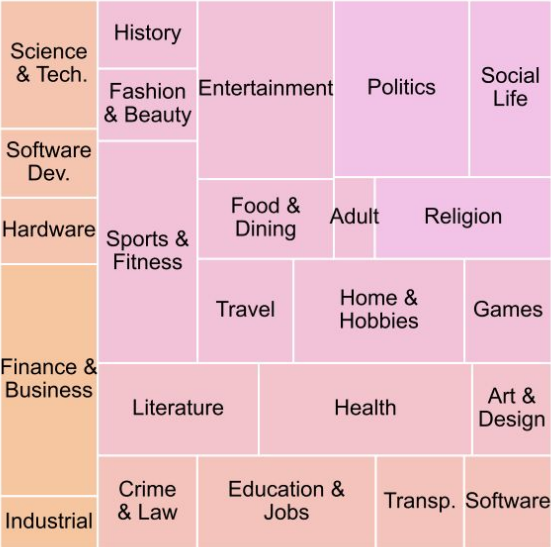


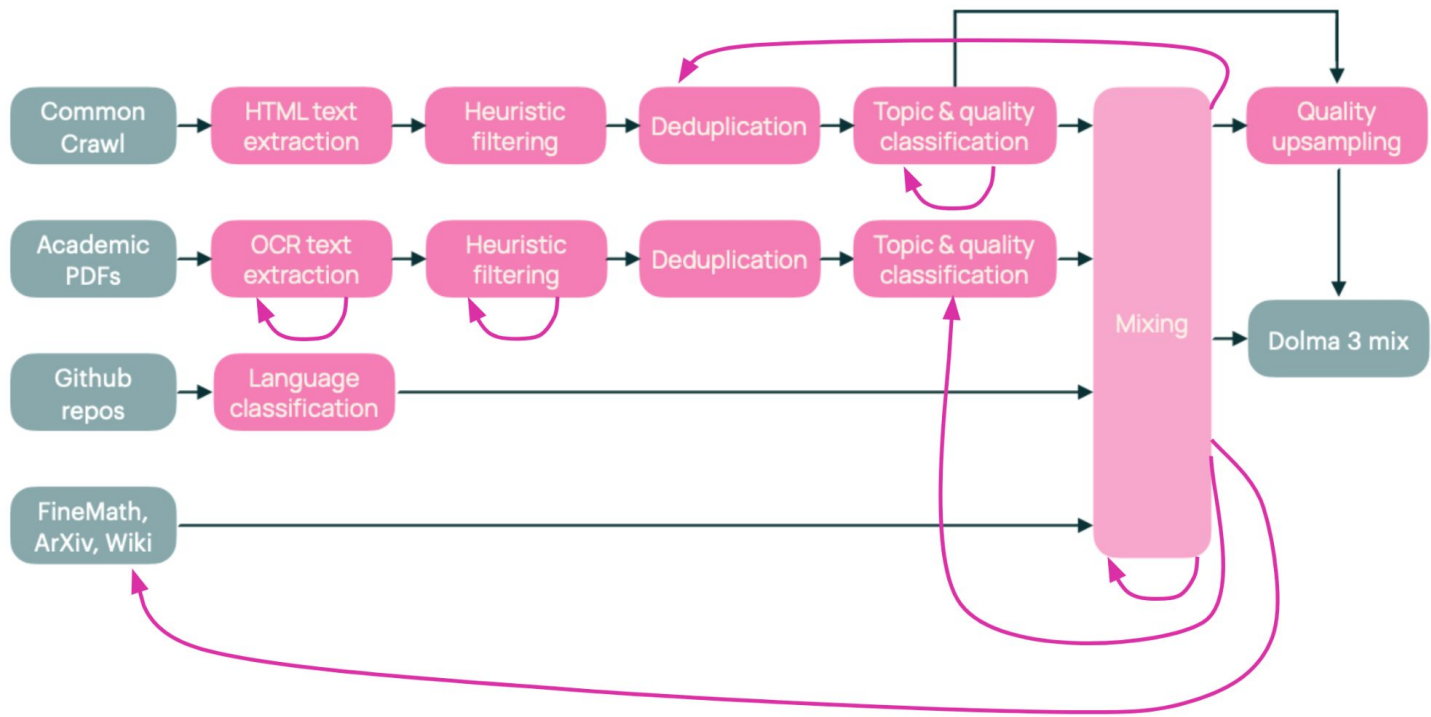
10 B Docs

Web Organizer

- Heuristic Filtering
- PII filtering
  - Language filtering
  - URL removal

- Deduplication
- Exact, global deduplication (new tooling!)
  - Min-hash (fuzzy)
  - Suffix array for boilerplate



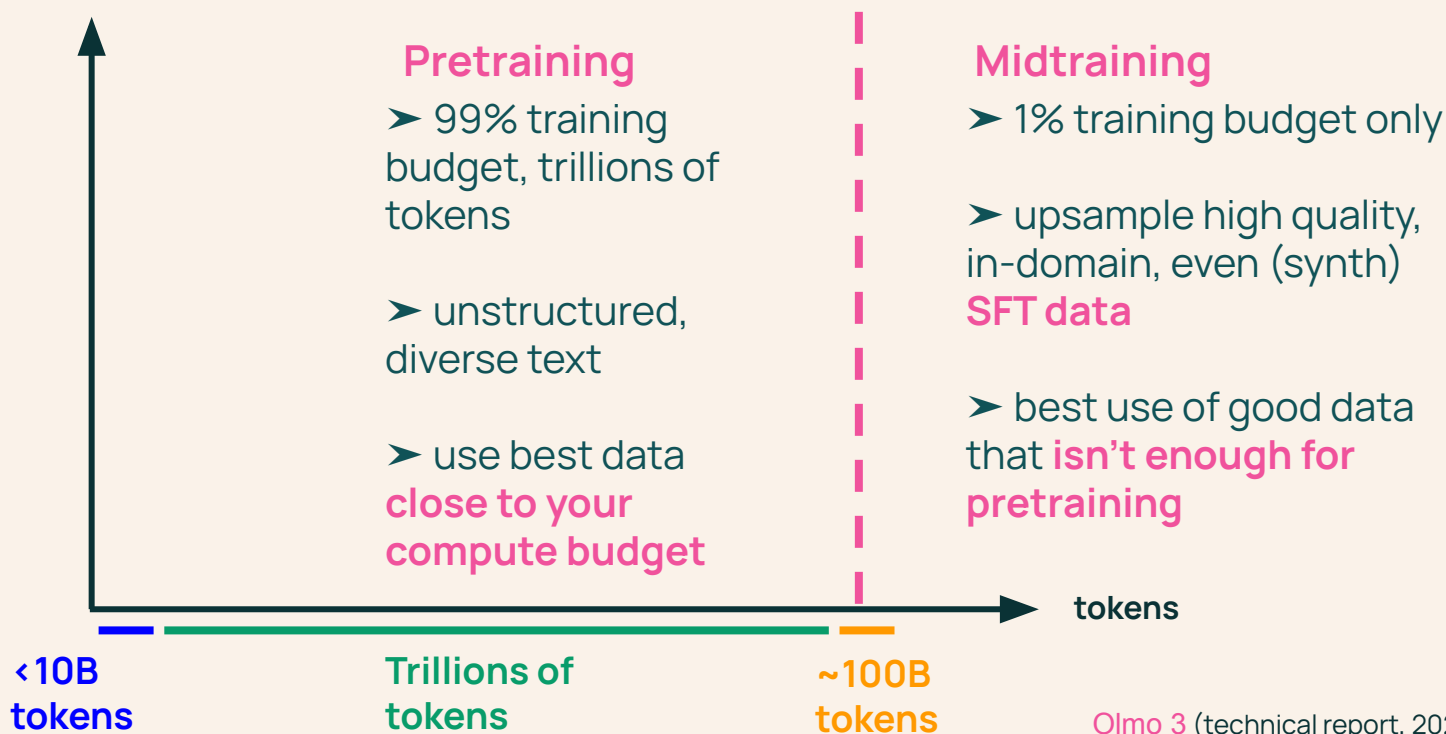


# Pretraining: Dolma 3

Source	Type	9T Pool		6T Mix	
		Tokens	Docs	Tokens	Docs
Common Crawl	Web pages	8.14T	9.67B	4.51T (76.1%)	3.15B
OLMOOCR science PDFs	Academic documents	972B	101M	805B (13.6%)	83.8M
Stack-Edu (Rebalanced)	GitHub code	137B	167M	409B (6.89%)	526M
arXiv	Papers with LaTeX	21.4B	3.95M	50.8B (0.86%)	9.10M
FineMath 3+	Math web pages	34.1B	21.4M	152B (2.56%)	95.5M
Wikipedia & Wikibooks	Encyclopedic	3.69B	6.67M	2.51B (0.04%)	4.24M
<b>Total</b>		<b>9.31T</b>	<b>9.97B</b>	<b>5.93T (100%)</b>	<b>3.87B</b>

Upsampled; higher quality increases probability of sampling a document

# Tension between Scale and Quality



# Dolma 3 **Dolmino**: Midtraining Mix

Math  
problem-solving  
through **code**  
and/or discussion

Instruction  
datasets

STEM PDFs and  
high-quality web  
pages

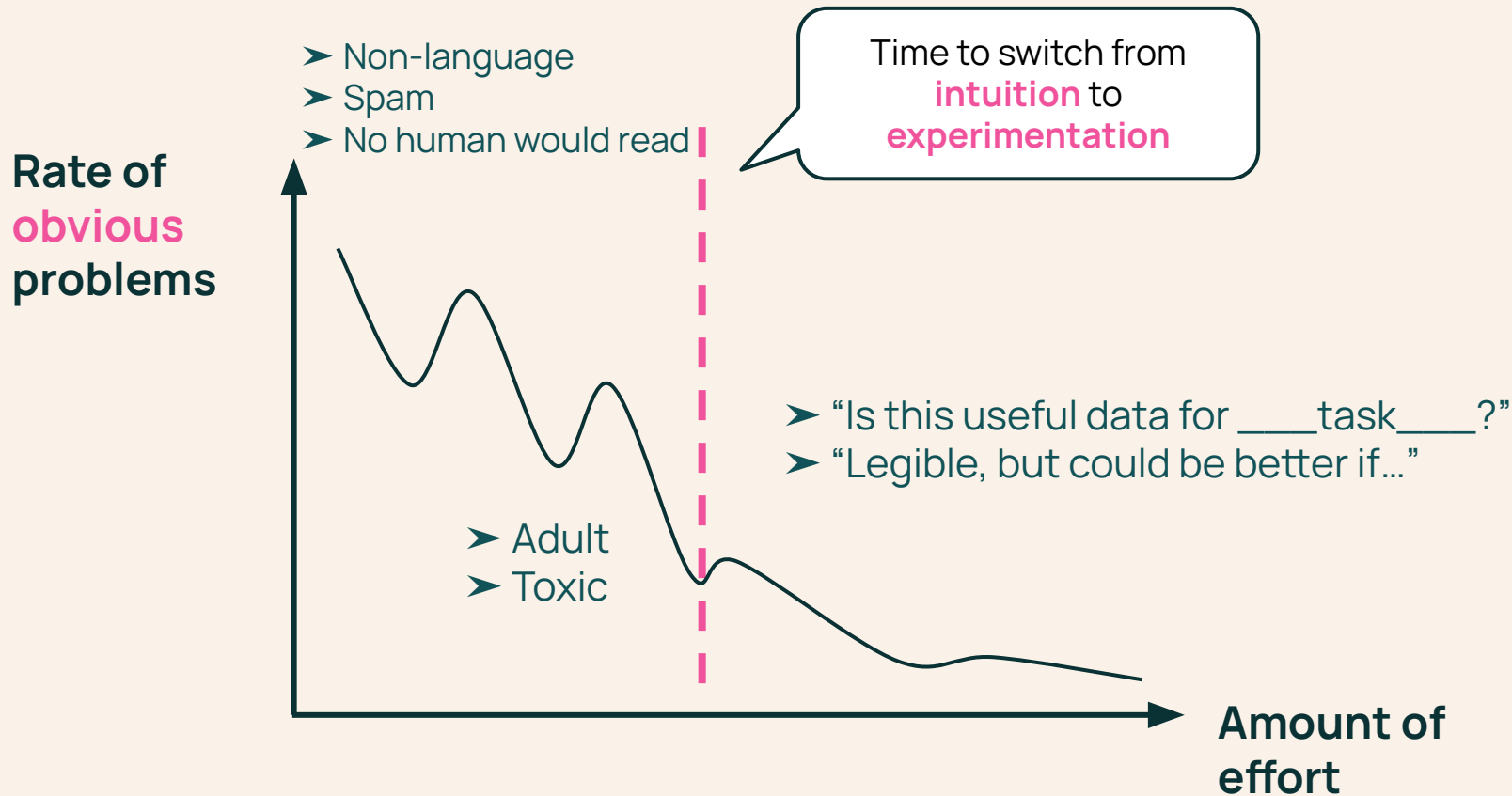


Type	Source	2T Pool		100B Mix	
		Tokens	Docs	Tokens	Docs
Math (synth)	TinyMATH Mind**	899M	1.42M	898M (0.9%)	1.52M
Math (synth)	TinyMATH PoT**	241M	729K	241M (0.24%)	758K
Math (synth)	CraneMath*	5.62B	6.55M	5.62B (5.63%)	7.24M
Math (synth)	MegaMatt*	3.88B	6.79M	1.73B (1.73%)	3.23M
Math (synth)	Dolmino Math^^	10.7B	21M	10.7B (10.7%)	22.3M
Code	StackEdu (FIM)^	21.4B	32M	10.0B (10.0%)	16.2M
Python (synth)	CraneCode*	18.8B	19.7M	10.0B (10.0%)	11.7M
QA (synth)	Reddit To Flashcards**	21.6B	370M	5.90B (5.9%)	101M
QA (synth)	Wiki To RCQA**	4.22B	22.3M	3.0B (3.0%)	16.3M
QA (synth)	Nemotron Synth QA^	487B	972M	5.0B (5.0%)	10.6M
Thinking (synth)	Math Meta-Reasoning**	1.05B	984K	381M (0.38%)	401K
Thinking (synth)	Code Meta-Reasoning**	1.27B	910K	459M (0.46%)	398K
Thinking (synth)	Program-Verifiable**	438M	384K	159M (0.16%)	158K
Thinking (synth)	OMR Rewrite FullThoughts^	850M	291K	850M (0.85%)	394K
Thinking (synth)	QWQ Reasoning Traces^	4.77B	438K	1.87B (1.87%)	401K
Thinking (synth)	General Reasoning Mix^	2.48B	668K	1.87B (1.87%)	732K
Thinking (synth)	Gemini Reasoning Traces^	246M	55.2K	246M (0.25%)	85.1K
Thinking (synth)	Llama Nemotron Reasoning Traces^	20.9B	3.91M	1.25B (1.25%)	368K
Thinking (synth)	OpenThoughts2 Reasoning Traces^	5.6B	1.11M	1.25B (1.25%)	402K
Instruction (synth)	Tulu 3 SFT^^	1.61B	1.95M	1.1B (1.1%)	1.45M
Instruction (synth)	Dolmino 1 Flan^^	16.8B	56.9M	5.0B (5.0%)	14.8M
PDFs	OLMOOCR Science PDFs (High Q.)^	240B	28.7M	4.99B (5.0%)	1.20M
Web pages	STEM-Heavy Crawl^	5.21B	5.16M	4.99B (5.0%)	5.53M
Web pages	Common Crawl (High Q.)^	1.32T	965M	22.4B (22.5%)	18.3M
<b>Total</b>		<b>2.19T</b>	<b>2.52B</b>	<b>99.95B (100%)</b>	<b>236M</b>

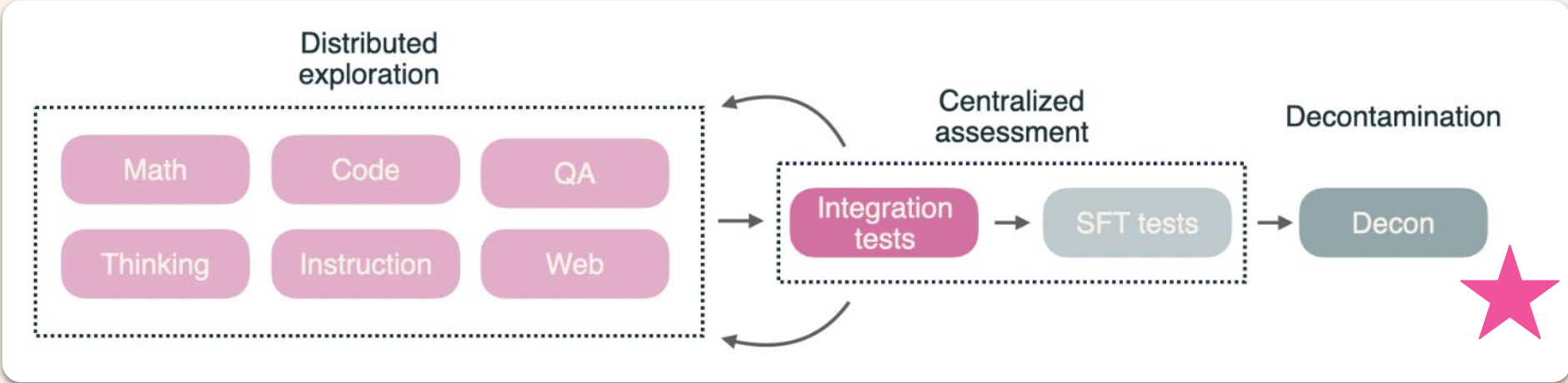
Demonstration of  
**diverse question  
structures**, rewritten  
from natural  
knowledge-rich data

**Math and code**  
problem-solving using  
**human-inspired  
meta-reasoning  
strategies/traces**

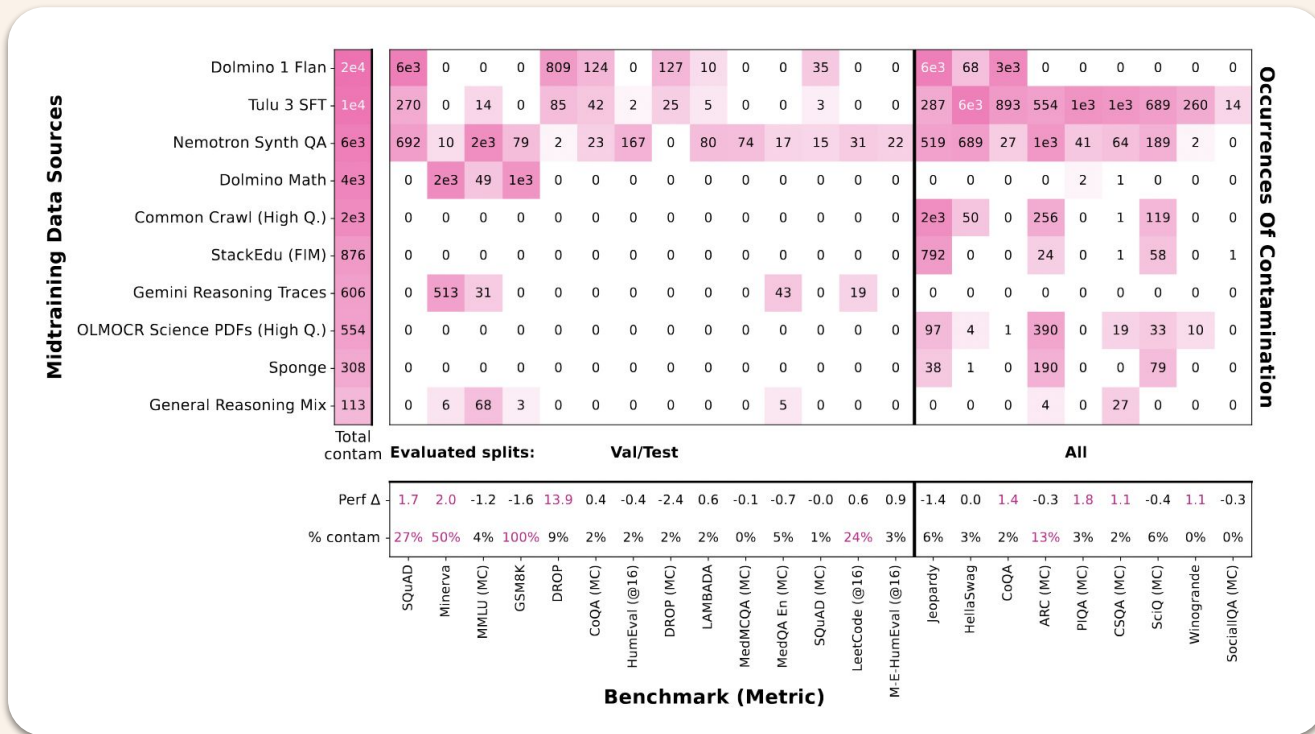
# Inspect Data Often, Favor Fast, Scalable Improvements

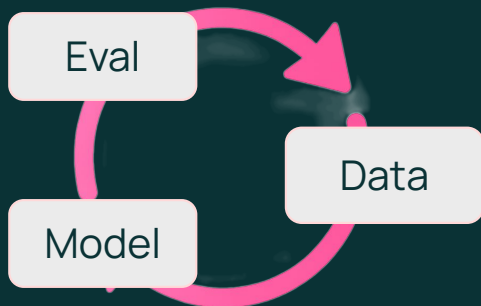


# Review: Map and Reduce Data Development



# Decontamination in Dolmino





## Olmo 3

Olmo Team\*

Allyson Ettlinger<sup>1</sup> Amanda Bertsch<sup>1,3</sup> Bailey Kuehl<sup>1,1</sup> David Graham<sup>1</sup>  
 David Heineman<sup>1</sup> Dirk Groeneveld<sup>1</sup> Faeze Brahman<sup>1</sup> Finbarr Timbers<sup>1</sup>  
 Hamish Ivison<sup>1,2</sup> Jacob Morrison<sup>1,2</sup> Jake Poznanski<sup>1</sup> Kyle Lo<sup>1,2</sup> Luca Soldaini<sup>1</sup>  
 Matt Jordan<sup>1</sup> Mayee Chen<sup>1,4</sup> Michael Noukhovitch<sup>1,5,6</sup> Nathan Lambert<sup>1</sup>  
 Pete Walsh<sup>1</sup> Pradeep Dasigi<sup>1</sup> Robert Berry<sup>1</sup> Robert Malik<sup>1</sup> Saurabh Shah<sup>1</sup>  
 Scott Geng<sup>1,2</sup> Shane Arora<sup>1</sup> Shashank Gupta<sup>1</sup> Taira Anderson<sup>1</sup> Teng Xiao<sup>1</sup>  
 Tyler Murray<sup>1</sup> Tyler Romero<sup>1</sup> Victoria Graf<sup>1,2</sup>

Akari Asai<sup>1,3</sup> Akshita Bhagia<sup>1</sup> Alexander Wettig<sup>1</sup> Alisa Liu<sup>2</sup> Aman Rangapur<sup>1</sup>  
 Chloe Anastasiades<sup>1</sup> Costa Huang<sup>1</sup> Dustin Schwenk<sup>1</sup> Harsh Trivedi<sup>1</sup> Ian Magnusson<sup>1,2</sup>  
 Jaron Lochner<sup>1</sup> Jiacheng Liu<sup>1</sup> Lester James V. Miranda<sup>1</sup> Maarten Sap<sup>1,3</sup> Malia Morgan<sup>1</sup>  
 Michael Schmitz<sup>1</sup> Michal Querquin<sup>1</sup> Michael Wilson<sup>1</sup> Regan Huff<sup>1</sup> Ronan Le Bras<sup>1</sup>  
 Rui Xin<sup>2</sup> Rulin Shao<sup>2</sup> Sam Sijmonsberg<sup>1</sup> Shannon Zhaijiang Shen<sup>1</sup> Shuyue Stella LF  
 Tucker Wilde<sup>1</sup> Valentina Pyatkin<sup>1</sup> Will Merrill<sup>1</sup> Yapei Chang<sup>2</sup> Yuling Gu<sup>1</sup> Zhiyuan Zeng<sup>1,2</sup>

Ashish Sabharwal<sup>1</sup> Luke Zettlemoyer<sup>2</sup> Pang Wei Koh<sup>1,2</sup>  
 Ali Farhadi<sup>1,2</sup> Noah A. Smith<sup>1,2</sup> Hannaneh Hajishirzi<sup>1,2</sup>

<sup>1</sup>Allen Institute for AI <sup>2</sup>University of Washington <sup>3</sup>Carnegie Mellon University <sup>4</sup>Stanford University <sup>5</sup>Mila  
<sup>6</sup>Université de Montréal <sup>7</sup>Princeton University <sup>8</sup>Massachusetts Institute of Technology <sup>9</sup>University of Maryland

\*OLMO 3 was a team effort; authors sorted alphabetically. \*marks core contributors. See author contributions here.

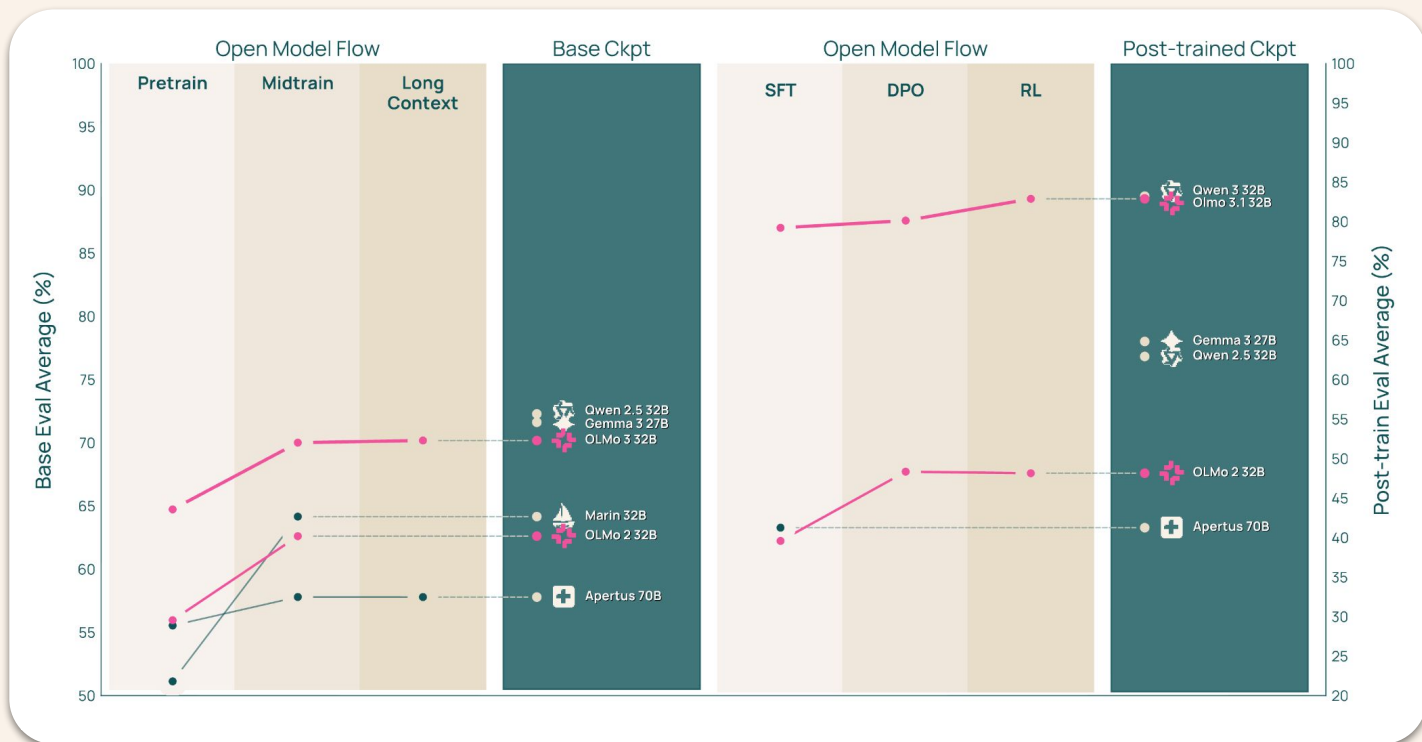
- 🔴 **Olmo 3 Base:** `Olmo-3-1025-7B` `Olmo-3-1125-32B`
- 🟡 **Olmo 3 Think:** `Olmo-3-7B-Think` `Olmo-3(12-1)-32B-Think`
- 🟠 **Olmo 3 Instruct:** `Olmo-3-7B-Instruct` `Olmo-3-1-32B-Instruct`
- 🟢 **Olmo 3 RL Zero:** `Olmo-3-7B-RL-Zero-(Math|Code|IF|General|Mix)` `Olmo-3-1-7B-RL-Zero-(Math|Code)`
- 🟣 **Base Data:** `Pretrain: Dolma 3 Mix` `Midtrain: Dolma 3 Dolmino Mix` `Long-ctx: Dolma 3 Longino Mix`
- 🟤 **Think Data:** `Dolci-Think-(SFT|DPO|RL)-7B` `Dolci-Think-(SFT|DPO|RL)-32B`
- 🟥 **Instruct Data:** `Dolci-Instruct-(SFT|DPO|RL)`
- 🟦 **RL-Zero Data:** `Dolci-RL-Zero-(Math|Code|IF|General)-7B` `Dolci-RL-Zero-Mix-7B`
- 🟧 **Training Code:** `olmo-core` (pretrain) `Open Instruct` (posttrain)
- 🟨 **Data Code:** `data-snap-rs` (data processing) `duplofocus` (deduplication) `dolma3` (data recipes)
- 🟩 **Eval Code:** `OLMOS` (eval suite) `lsecm` (eval decontamination)
- 🟪 **Training Logs:** `Olmo-3-7B-(Base|Think|Instruct|RL-Zero)` `Olmo-3-32B-(Base|Think|Instruct)`
- 🟫 **Demo:** `32B Think` `32B Instruct` `7B Think` `7B Instruct`
- 🟬 **Contact:** `olmo@allenai.org`

### Abstract



We introduce **OLMO 3**, a family of state-of-the-art, fully-open language models at the 7B and 32B parameter scales. OLMO 3 model construction targets long-context reasoning, function calling, coding, instruction following, general chat, and knowledge recall. This release includes the entire model flow, i.e., the full lifecycle of the family of models, including every stage, checkpoint, data point, and dependency used to build it. Our flagship model, **OLMO 3.1 THINK 32B**, is the strongest fully-open thinking model released to-date.

# Olmo 3 Model Flow

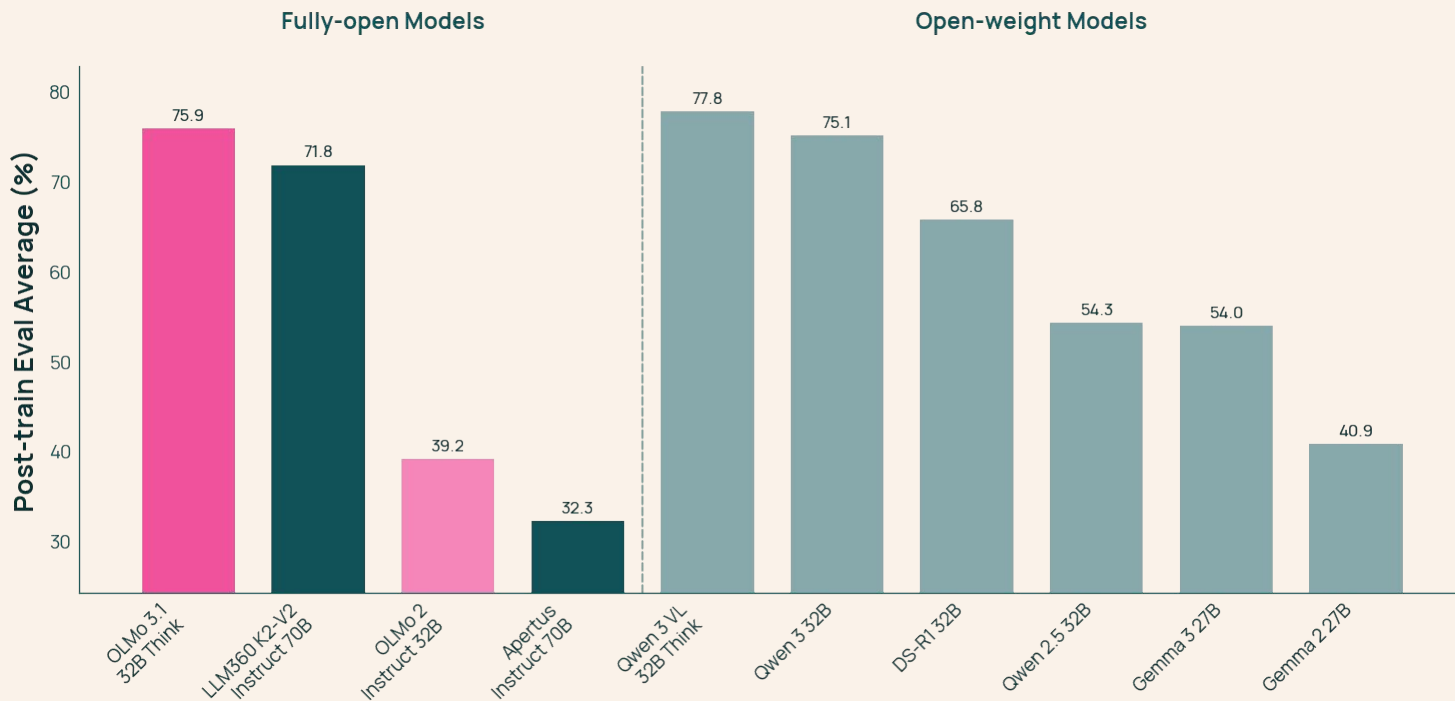




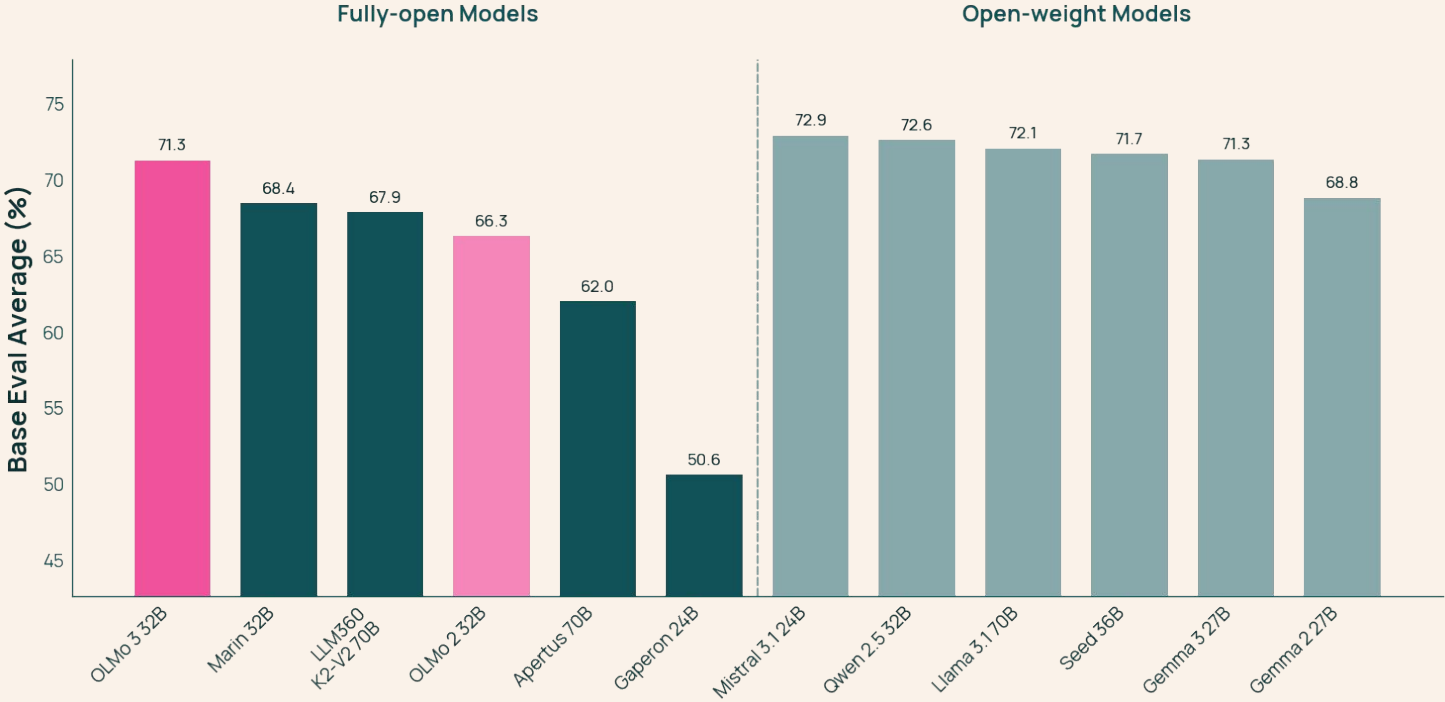


[extra slides below]

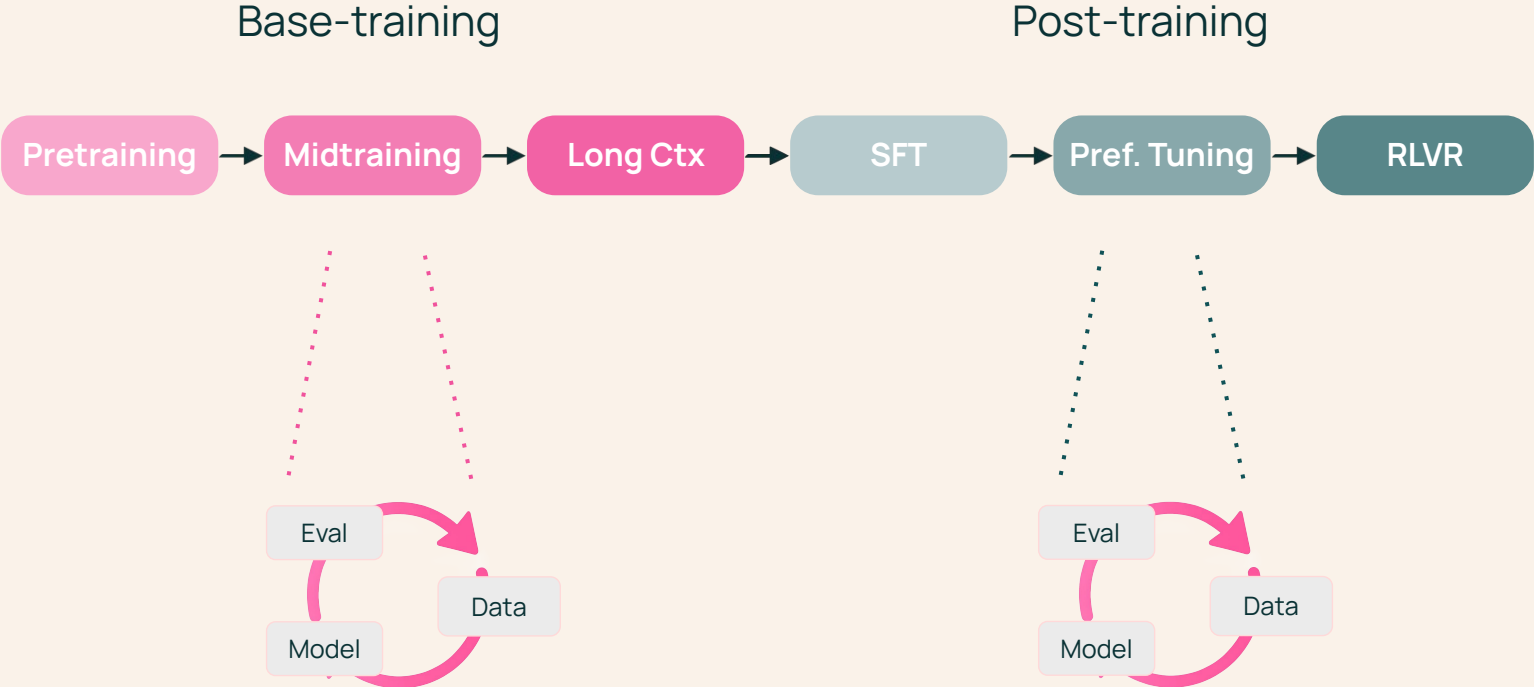
# Olmo 3 Model Flow



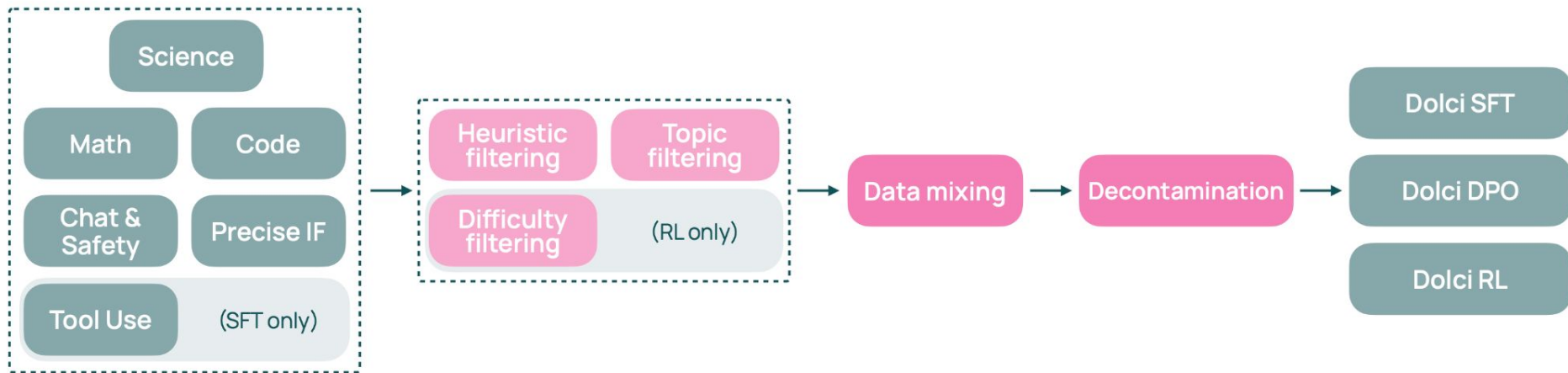
# Olmo 3 Model Flow



# Olmo 3 Model Flow



# Posttrain for Olmo 3 Think



## SFT Data

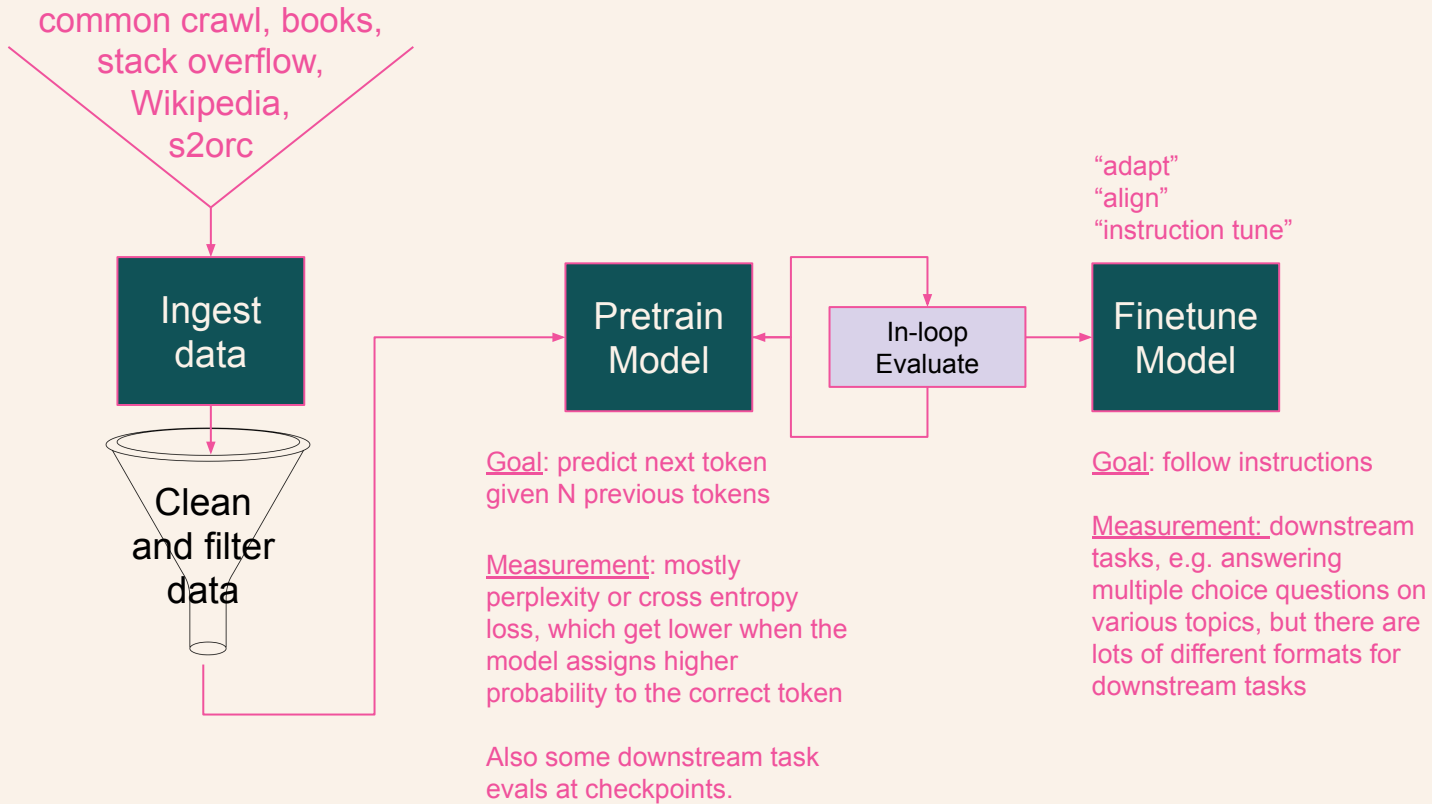
Modular data  
curation & mixing

## Preference data

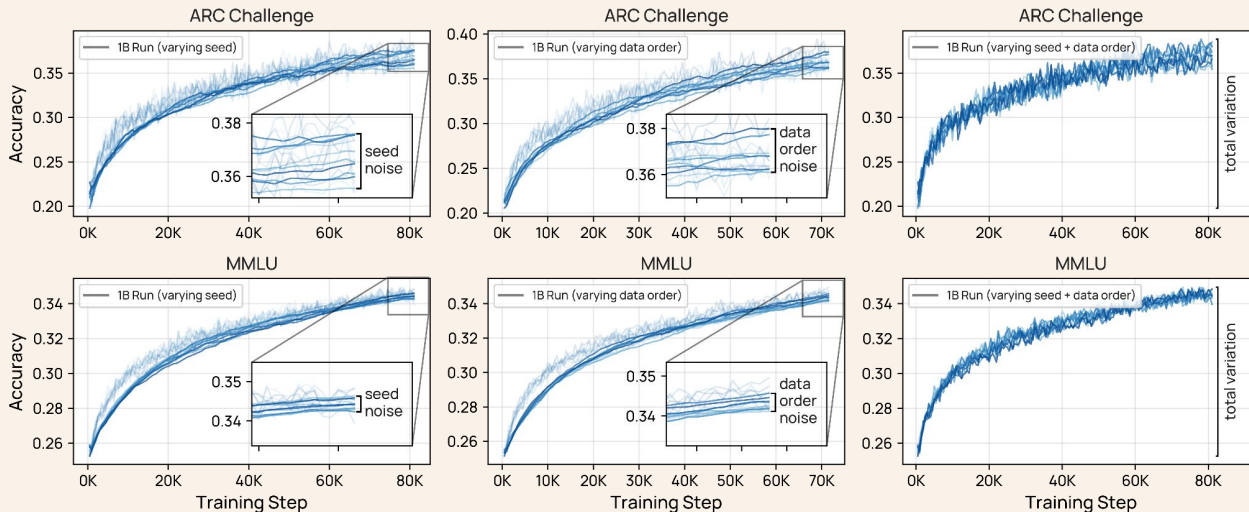
Learn from contrast  
between data points

## Scaled up RL

Enable thinking &  
adaptive RL

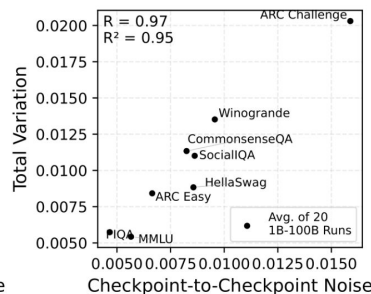
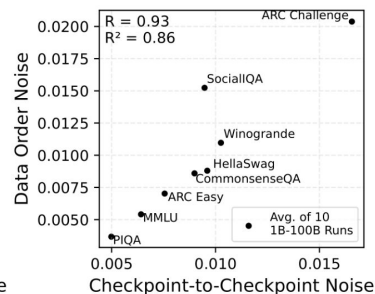
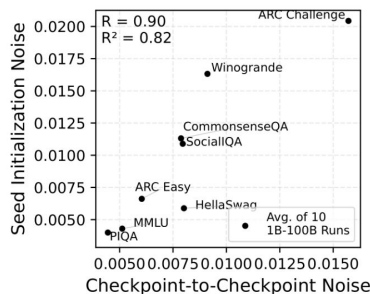


# Many sources of noise ...



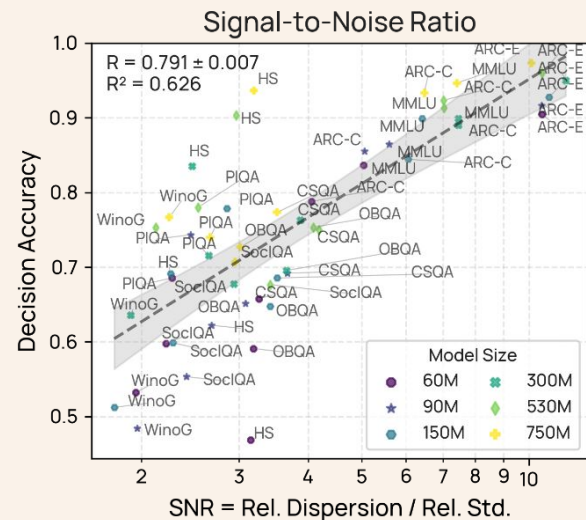
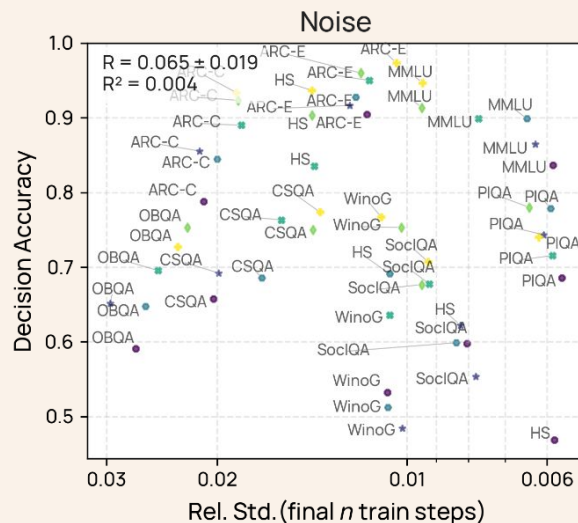
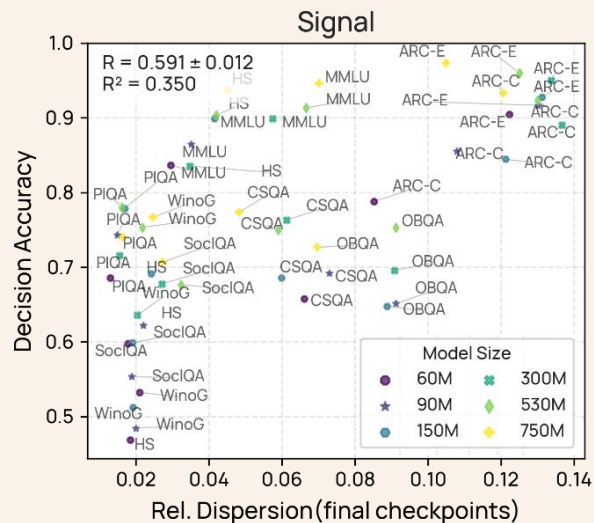
10 1B-5xC models varying:  
data order and  
random seed

... seed noise  
correlates with  
step-to-step noise!



+ Total variation  
correlates w/  
step-to-step noise

# Only **signal** or **noise** alone do not explain rank agreement from small to large scale... .. but the **signal-to-noise ratio** does!



# Problem 1: No “standard” config

## Swarm construction:

*RQ1* What is the smallest proxy model size (the number of parameters,  $S_{\text{small}}$ ) such that decision-making generalizes to larger target models?

*RQ2* How many proxy runs  $K$  do we need to learn a good mix on  $m$  domains?

*RQ3* How should we specify the distribution  $\mathcal{P}$  to sample the mixes for the proxy runs?

## Regression model:

*RQ4* Is there an optimal family of regression models ( $\mathcal{F}$ ) for predicting mix performance?

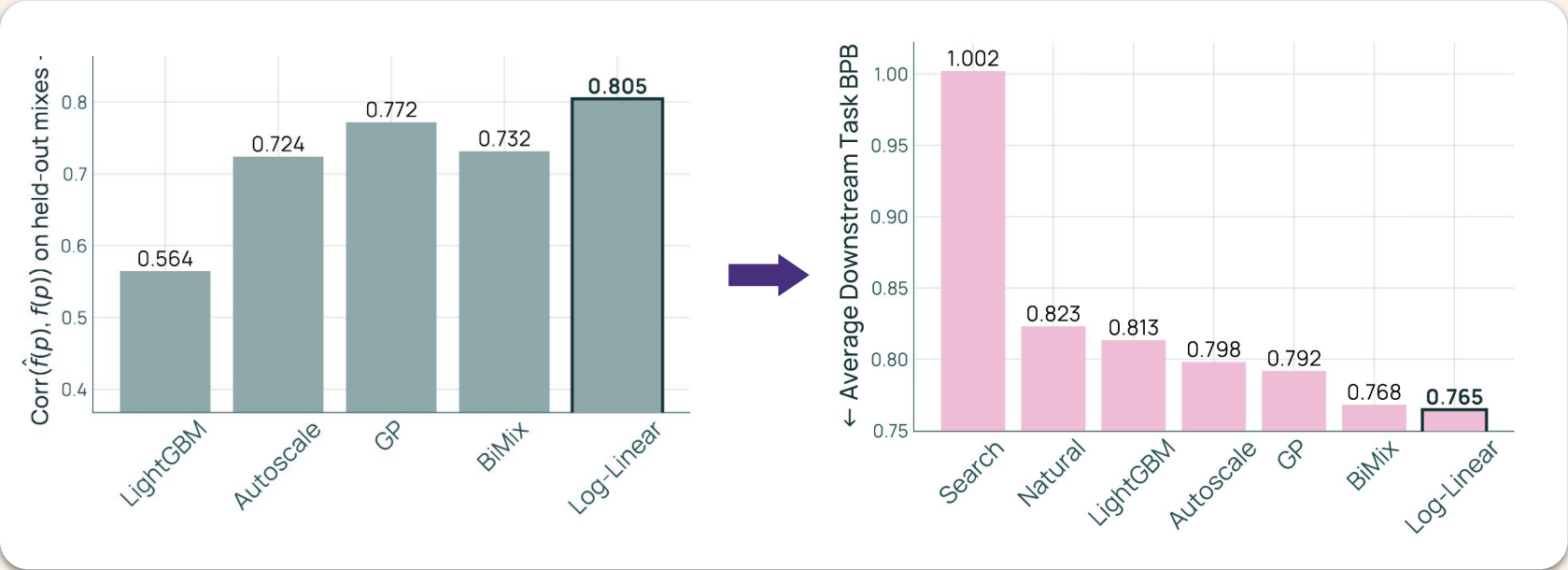
*RQ5* At what granularity should we fit the regression models in order to construct  $\hat{f}(p)$ ?

## Mix optimization:

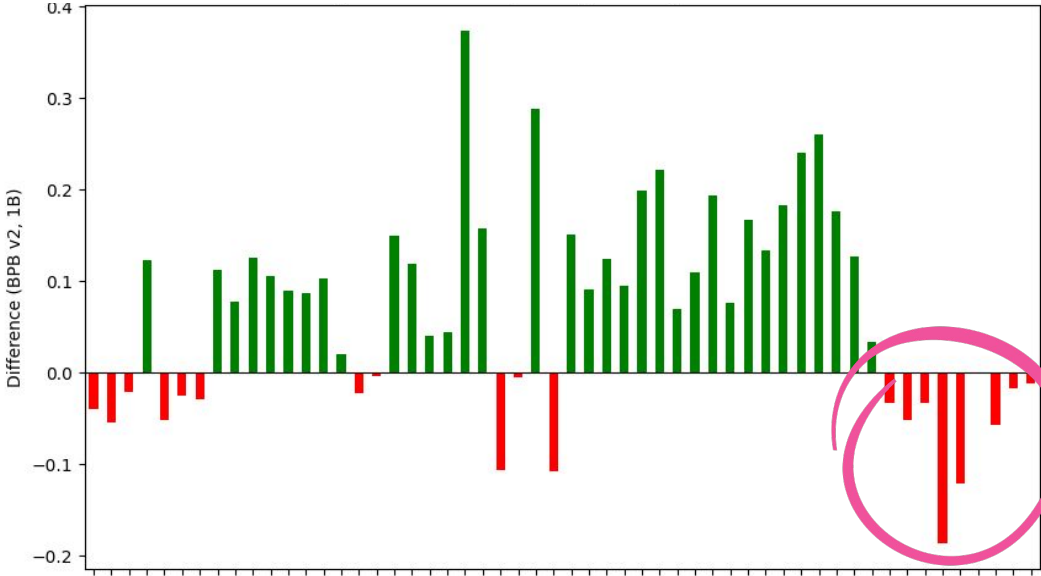
*RQ6* How do we mix under finite data constraints?

*RQ7* How do we solve the optimization problem?

# Finding 3a: Regression fit is decently indicative of downstream performance

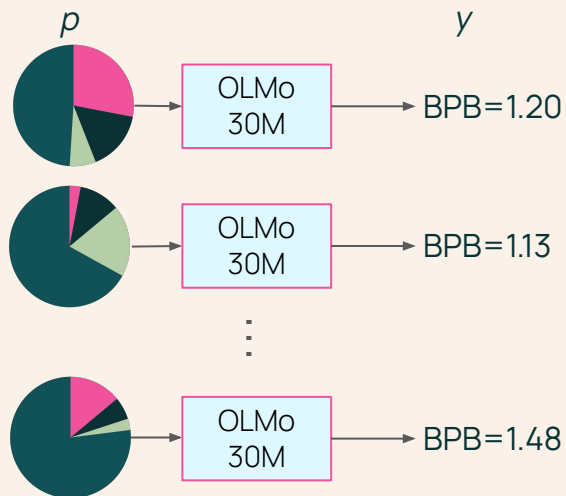


# Data mixing: Operations

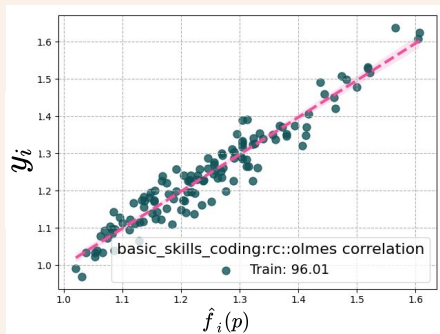


# From Pools to a Mix

1. **Swarm runs**: train  $K$  30M models with randomly sampled mixtures  $p$

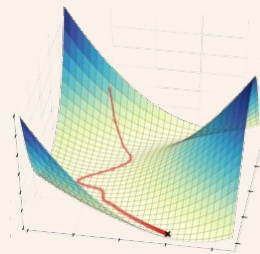
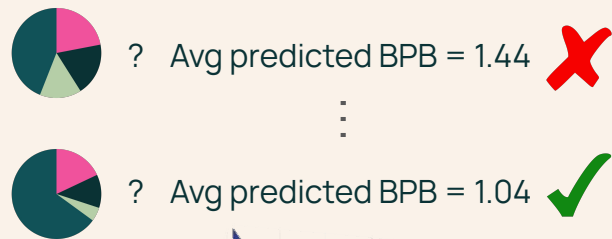


2. **Fit a regression model** for each benchmark task  $\hat{f}_i(p) \approx y_i$

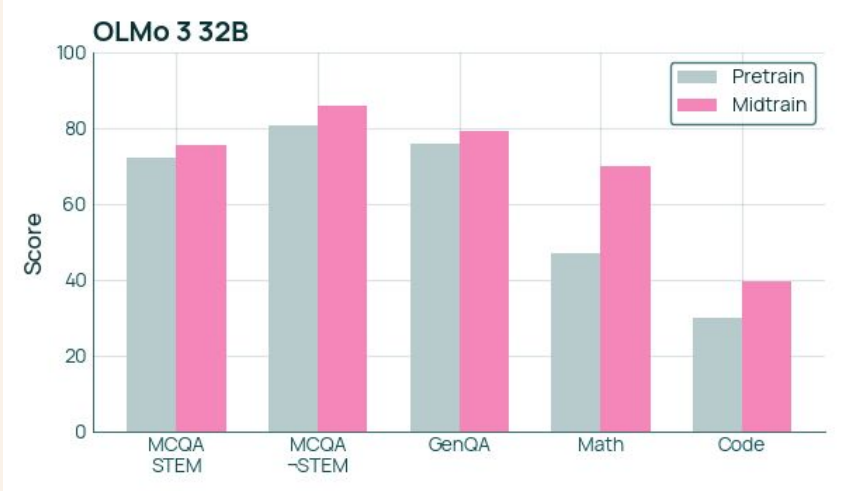
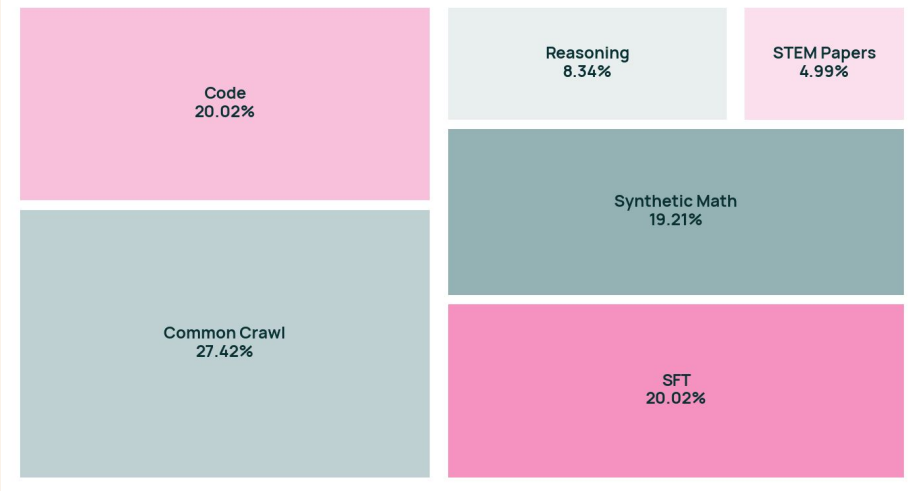


3. **Solve optimization problem** to get optimal mix  $p^*$

$$\text{minimize}_{p \in \Delta^{m-1}} \frac{1}{n} \sum_{i=1}^n \hat{f}_i(p)$$



# Midtraining Olmo 3



Boost data that closely match post-train capability targets

# Finding 5: Historical data can be optimally reused to minimize experimental cost over time

## Swarm

$$p_1 = [0.3, 0.7]$$

$$p_2 = [0.4, 0.6]$$

$$p_3 = [0.8, 0.2]$$

...

$$p_N = [0.1, 0.9]$$



$$p^* = [0.66, 0.34]$$



# Finding 5: Historical data can be optimally reused to minimize experimental cost over time

**Swarm**

$$p_1 = [0.3, 0.7]$$

$$p_2 = [0.4, 0.6]$$

$$p_3 = [0.8, 0.2]$$

...

$$p_N = [0.1, 0.9]$$

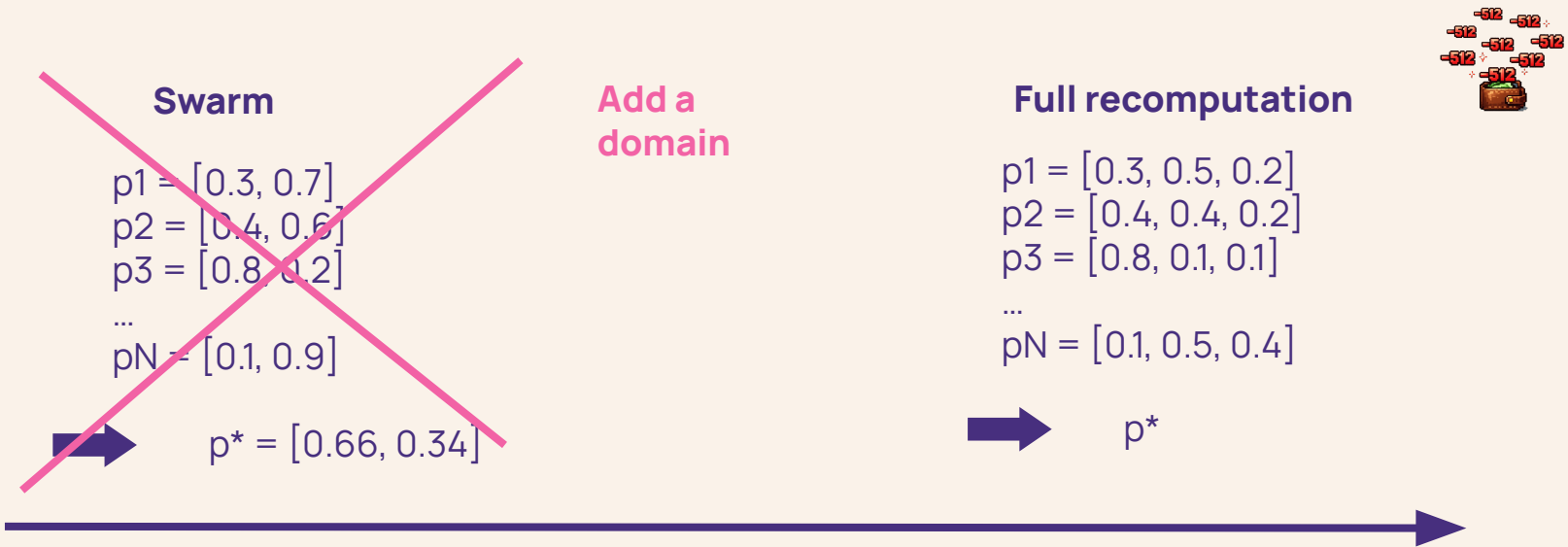
**Add a  
domain**



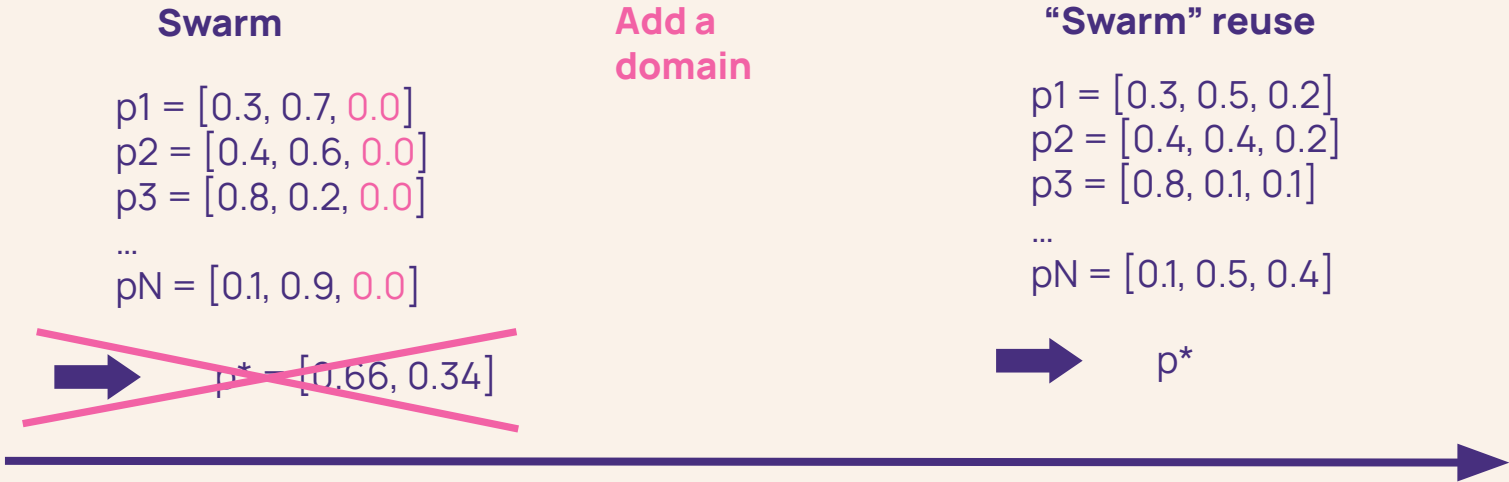
$$p^* = [0.66, 0.34]$$



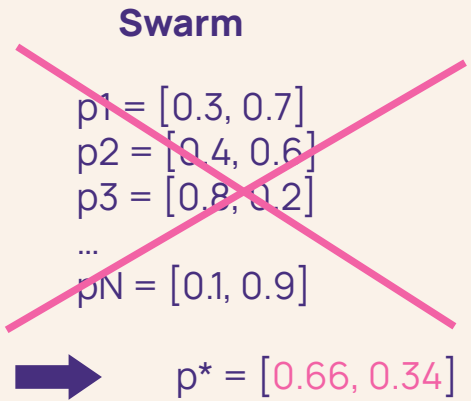
# Finding 5: Historical data can be optimally reused to minimize experimental cost over time



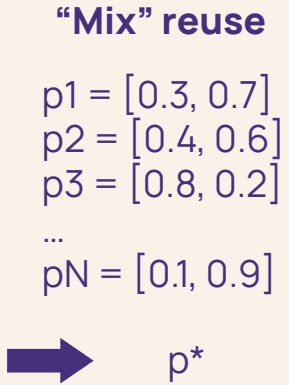
# Finding 5: Historical data can be optimally reused to minimize experimental cost over time



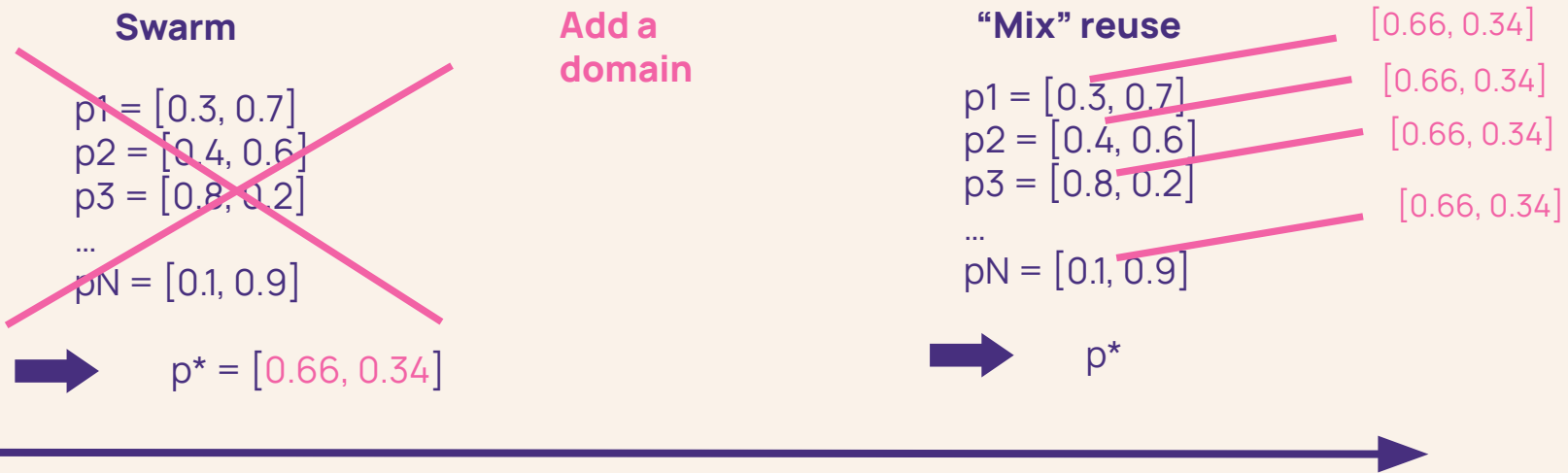
# Finding 5: Historical data can be optimally reused to minimize experimental cost over time



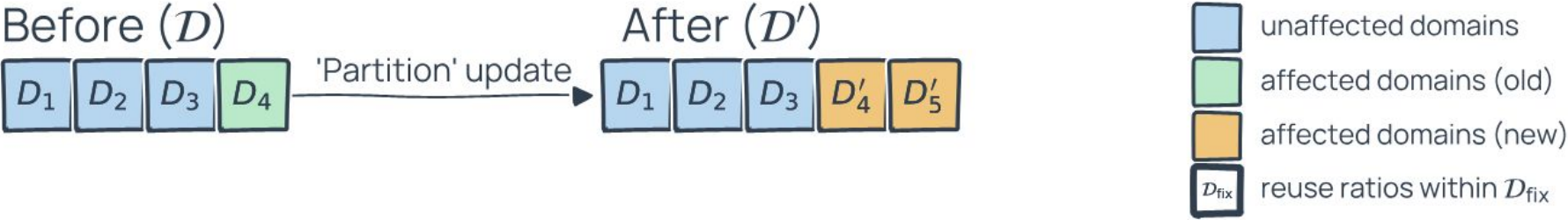
Add a domain



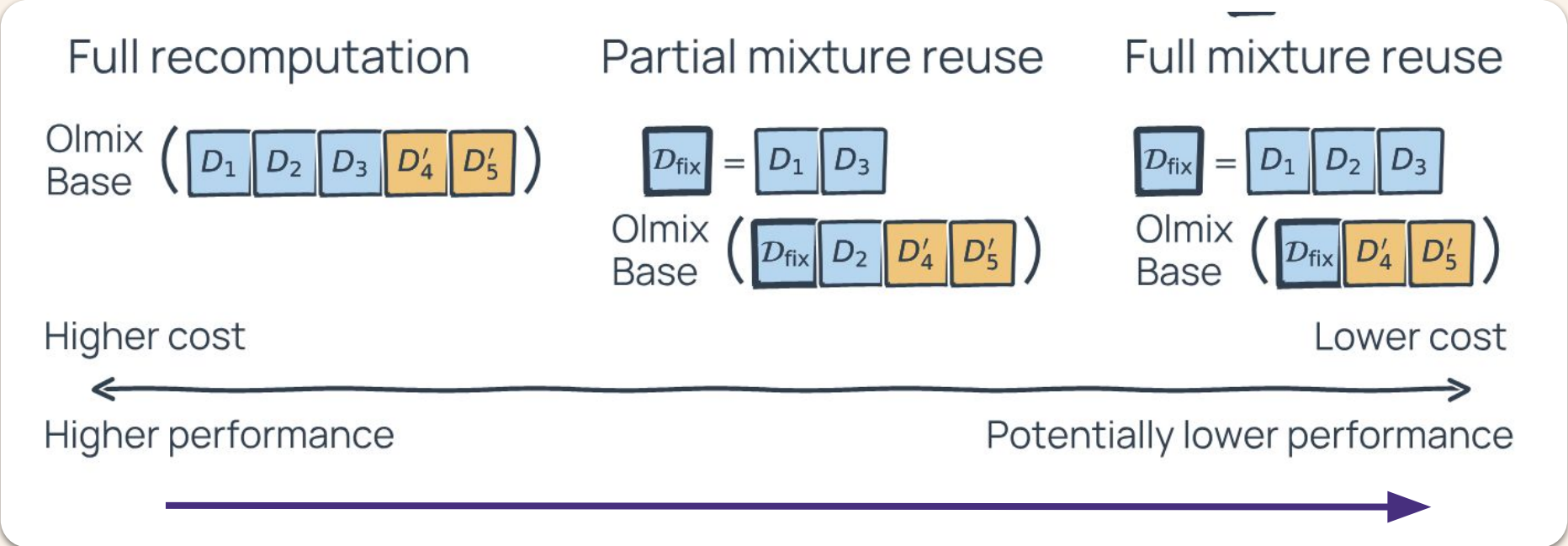
# Finding 5: Historical data can be optimally reused to minimize experimental cost over time



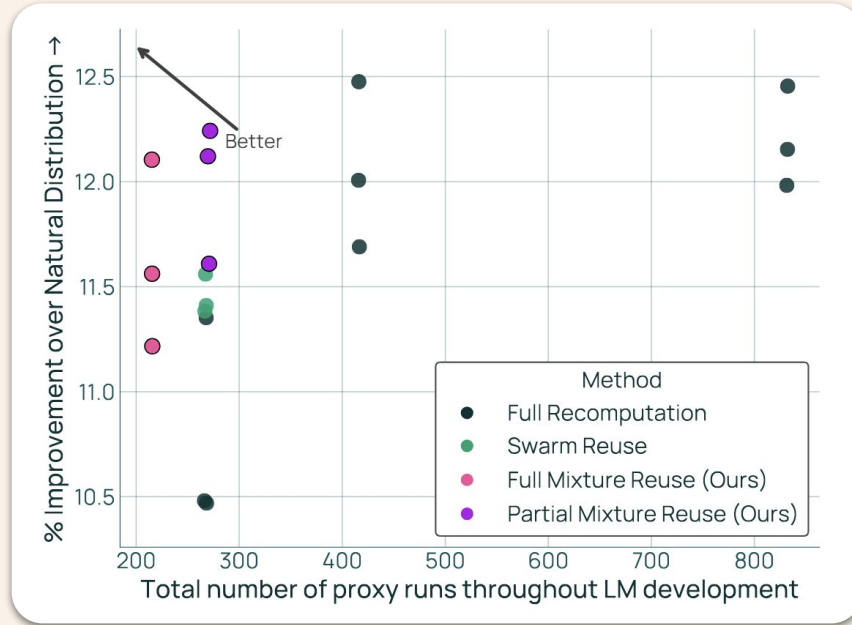
# Finding 5: Mixture reuse



# Finding 5: Mixture reuse



# Finding 5: Historical data can be optimally reused to minimize experimental cost over time



## Finding 5: Historical data can be optimally reused to minimize experimental cost over time

**Theorem 4.1** (Performance gap bound). *The performance gap is bounded by*

$$F(q^*(\tilde{p}_{\mathcal{D}_{fix}})) - F(q^*) \leq C \|\tilde{p}_{\mathcal{D}_{fix}} - q_{\mathcal{D}_{fix}}^*\|,$$
$$C = \mathcal{E}(\|\tilde{p}_{\mathcal{D}_{fix}} - q_{\mathcal{D}_{fix}}^*\|)(\kappa(\alpha_{fix}, \alpha_{comp}) + \|\alpha_{fix}\|)$$

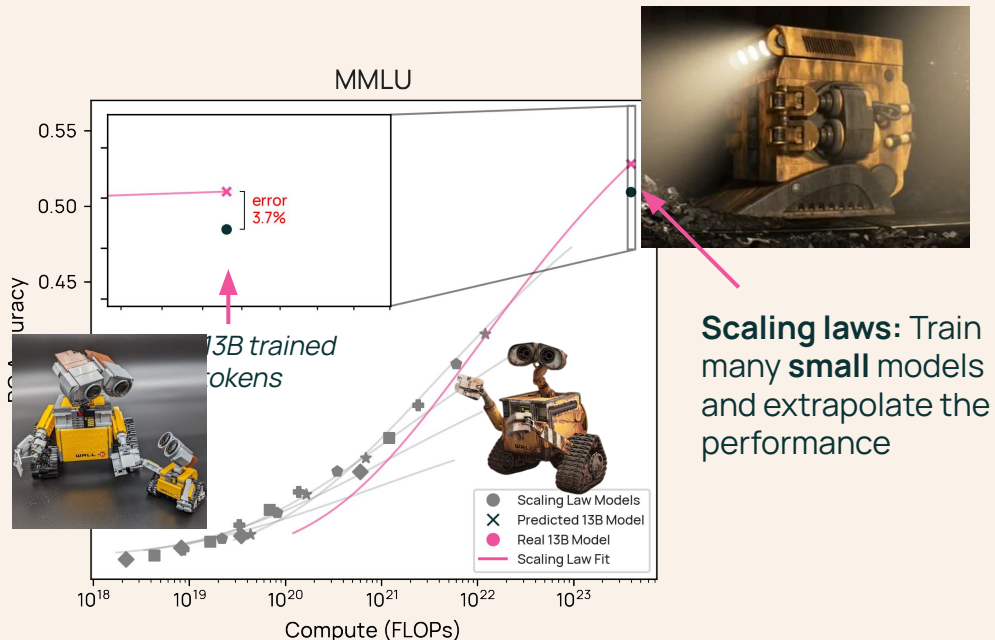
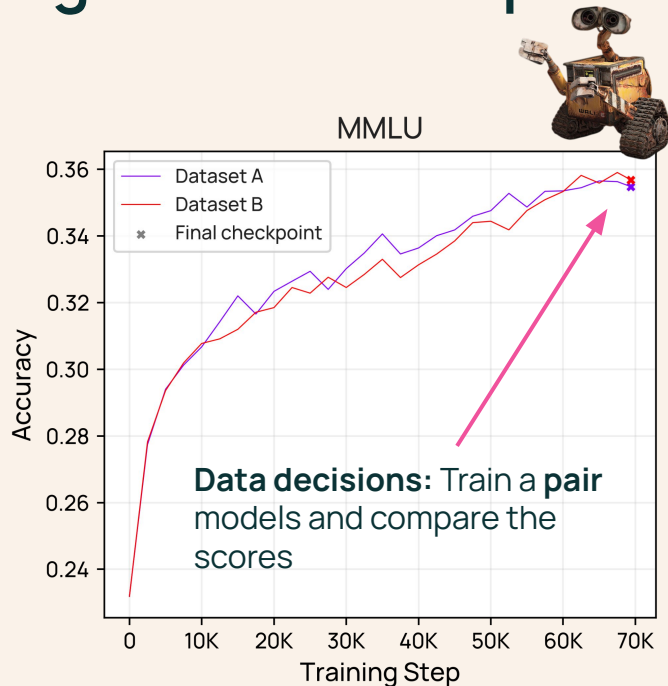
**Theorem 4.2.** *Assume that  $\tilde{p}$  is the solution to (2) on  $\mathcal{D}$ . When new domains are added, the reuse gap is bounded by*

$$\|\tilde{p}_{\mathcal{D}_{fix}} - q_{\mathcal{D}_{fix}}^*\| \leq \mathcal{E}(1 - \rho^*)\kappa(\alpha_{fix}, \alpha_{comp})(1 - \rho^*)$$

# Problem 1: No “standard” config

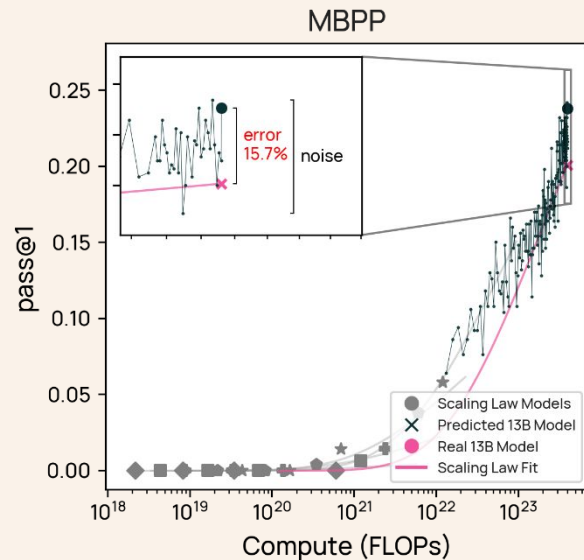
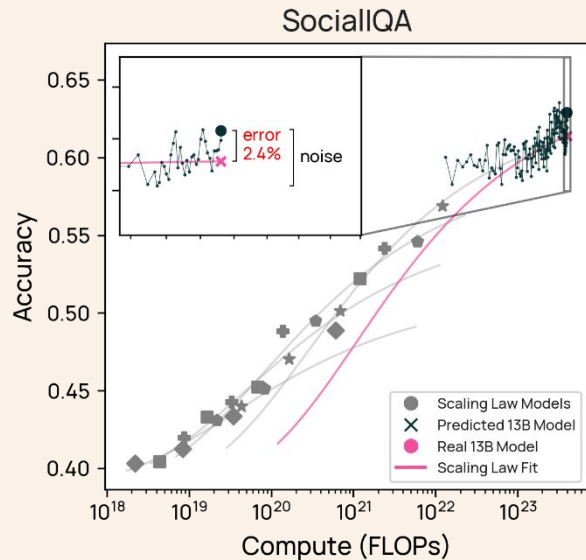
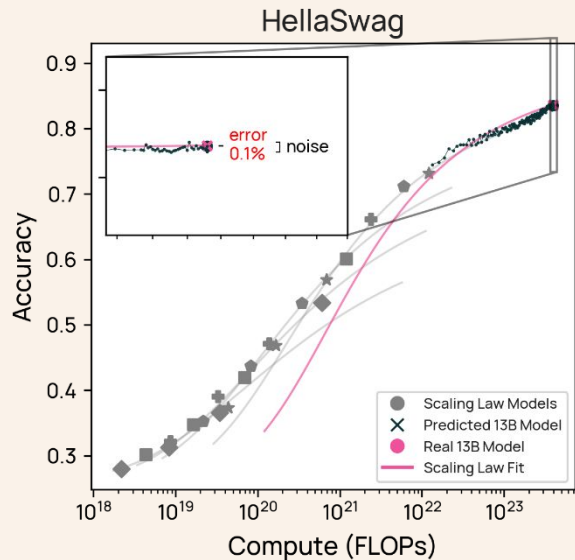
Design Choice	RegMix (Liu et al., 2025a)	DML (Ye et al., 2025)	AutoScale (Kang et al., 2025)	BiMix (Ge et al., 2025b)	ADMIRE-BayesOpt (Chen et al., 2025b)	CLIMB (Diao et al., 2025)
<b>Swarm Construction</b>						
Proxy model size	1M	70, 160, 305, 410M	Target	280M	1M, 60M	350M
Swarm size (vs $m$ domains)	512 ( $m = 17$ )	20 ( $m = 7$ )	$2m + 1$	4	101 ( $m = 17$ )	112 ( $m = 21$ )

# Making decisions in pretraining == extrapolating eval

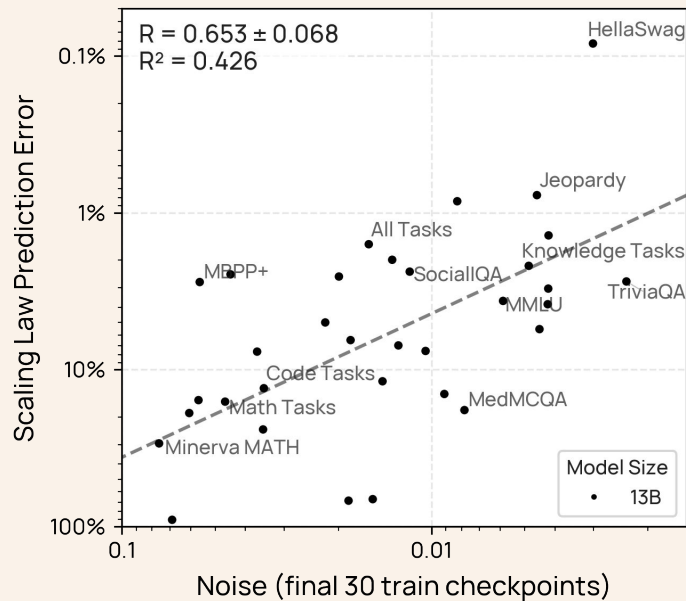


DataDecide: How to Predict Best Pretraining Data with Small Experiments (ICML, 2025)  
Establishing Task Scaling Laws via Compute-Efficient Model Ladders (COLM, 2025)

# Predicting task performance using scaling laws is sensitive to noise!

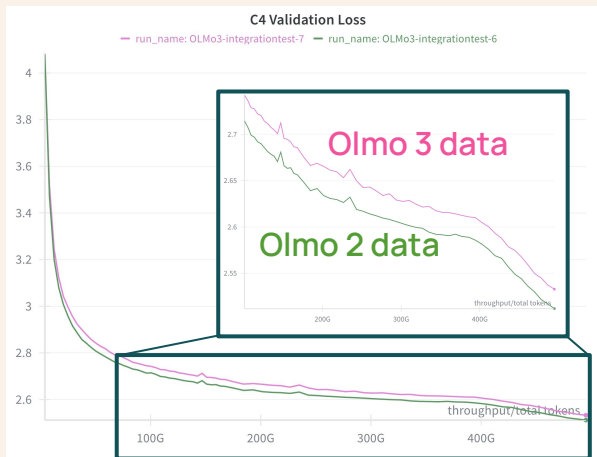


# Predicting task performance using scaling laws is sensitive to noise!

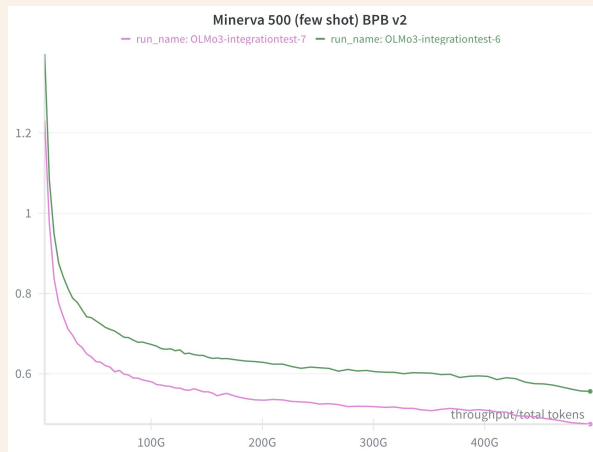


## ② Small-scale metrics:

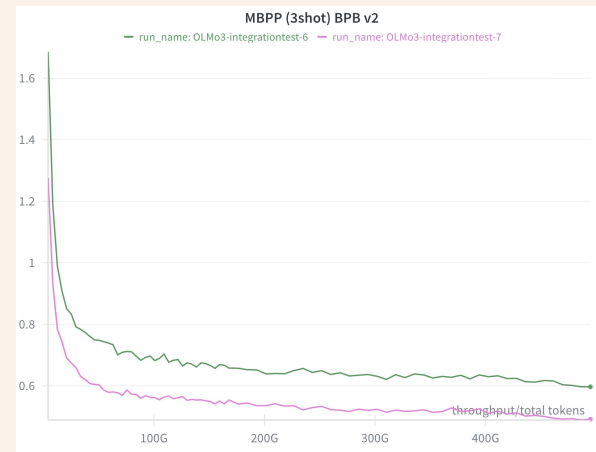
### C4 Loss



### Math 500 BPB



### MBPP BPB (Python Coding)



At 7B 500B tokens, **Olmo 3 data** has a worse fit than **Olmo 2 data** on C4 loss, but better on math, code text

## ② Small-scale metrics:

### C4 Loss

