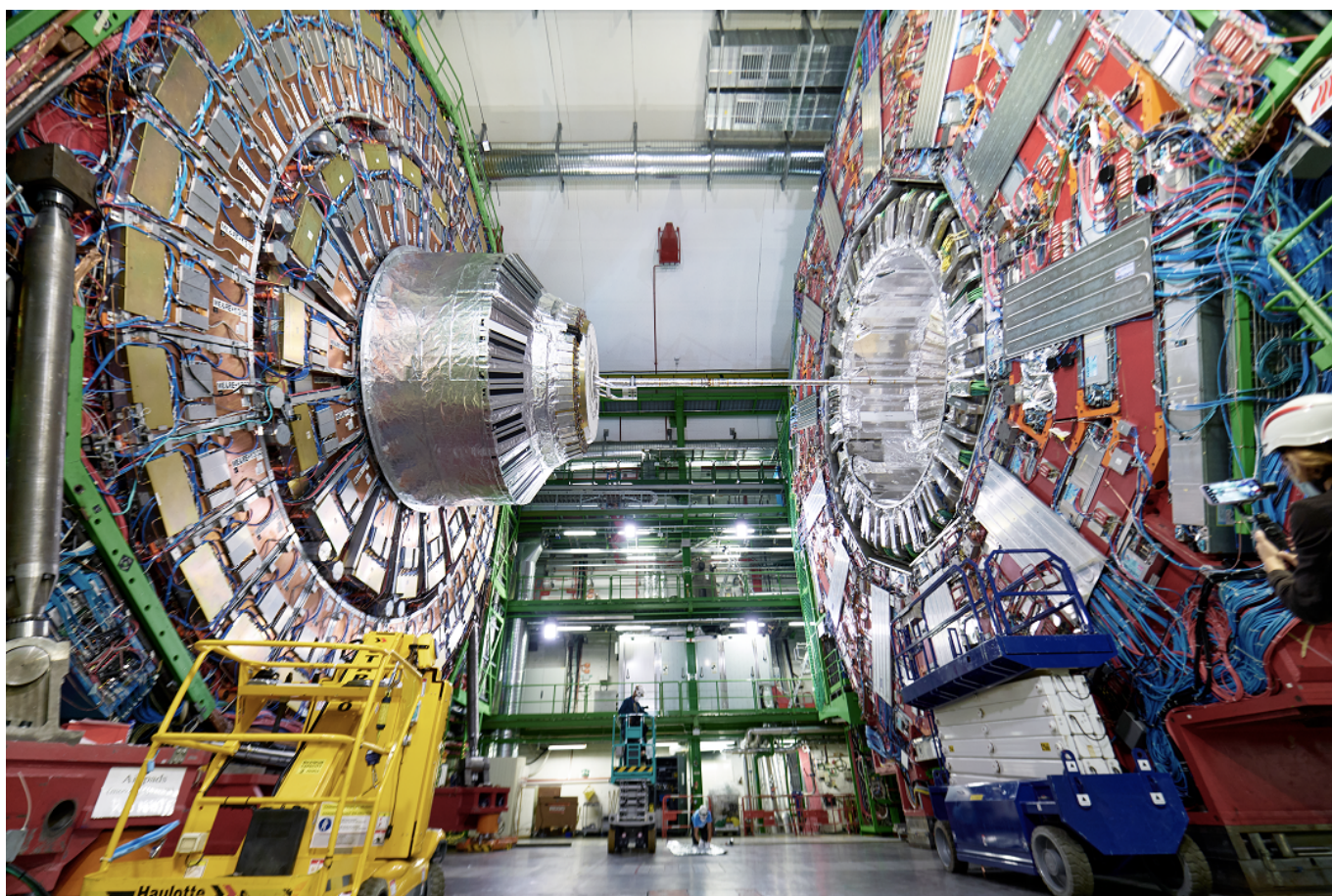


Ultrasnelle machine learning bij CMS

Machine learning, ook wel AI, is in de hele maatschappij hét grote gespreksonderwerp van het moment. Ook binnen de deeltjesfysica wordt machine learning gebruikt bij steeds meer onderdelen van de analyse. Recent werk richt zich op de toepassing van machine learning helemaal aan het begin van de wetenschappelijke keten, bij de real-time dataverwerking van deeltjesdetector CMS.



Afbeelding 1. De CMS-detector. Credits: CERN Samuel, Joseph Hertzog (CERN).

Onder de CERN-campus in Genève kan je de op dit moment grootste deeltjesversneller ter wereld vinden: de Large Hadron Collider (LHC). In de 27 kilometer lange, cirkelvormige tunnel van de LHC botsen protonen met enorme snelheden op elkaar. De brokstukken worden geanalyseerd in diverse meetstations, waaronder CMS, de Compact Muon Solenoid-detector -

zie afbeelding 1. De enorme hoeveelheden data die deze experimenten te verwerken krijgen, hebben al geleid tot IT-innovaties zoals cloud computing. Zowel de LHC als CMS zullen binnenkort uitgebreide upgrades ondergaan, waardoor verdere innovatie een vereiste is.

De data-uitdaging

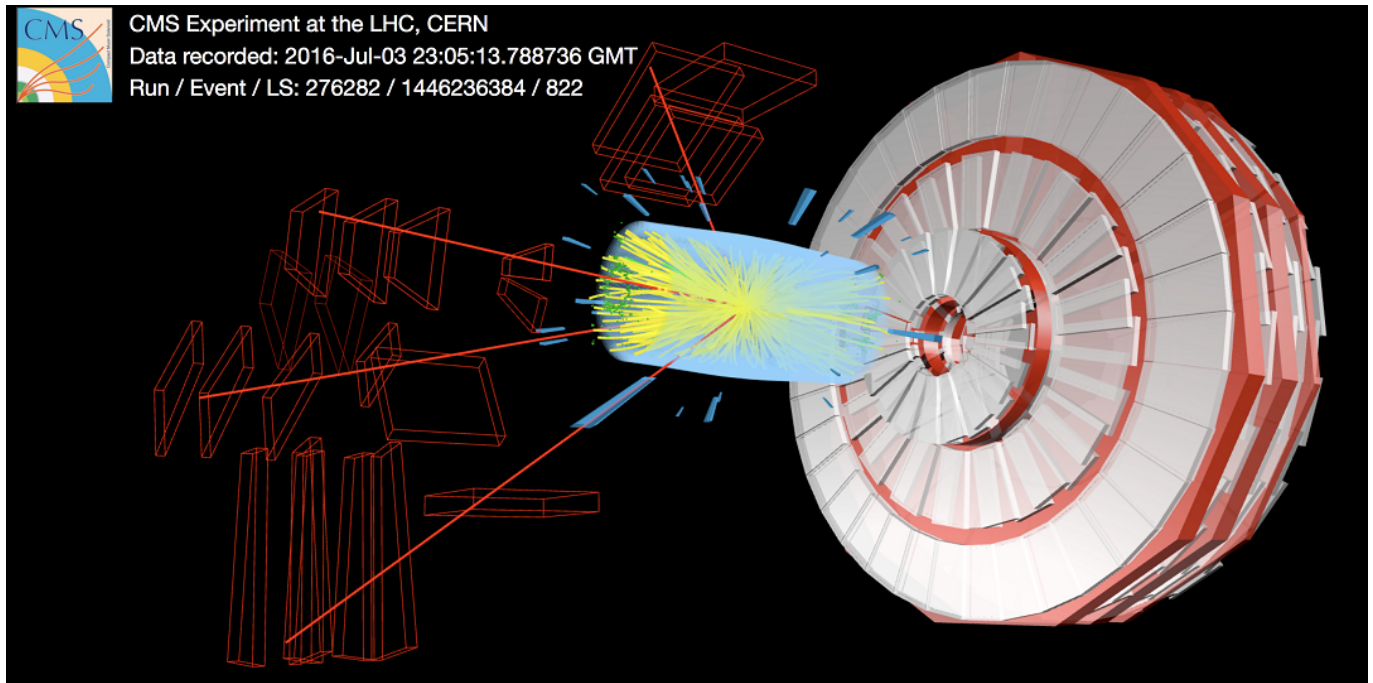
Laten we beginnen met het concreter maken van de probleemstelling: met wat voor datastromen heb je te maken bij LHC-experimenten zoals het CMS-experiment? Wanneer de LHC onder normale omstandigheden opereert, is de versneller gevuld met zo'n 2500 bundels van protonen, met ongeveer honderd miljard protonen per bundel. Er zit een pauze van 25 nanoseconden (ns) tussen iedere twee bundels, wat wil zeggen dat er bij een experiment als CMS 28 miljoen keer per seconde een bundel langskomt. Doe dit in beide richtingen en je hebt 28 miljoen botsingen tussen bundels per seconde, waarbij er bij elk van die botsingen ongeveer 60 individuele botsingen tussen protonen plaatsvinden. Bij iedere protonenbotsing wordt er 1 MB aan data gegenereerd door de detector. In totaal wordt er dus 1 PB (petabyte, 10^{15} bytes) per seconde aan data gegenereerd. Dat is 1% van het wereldwijde internetverkeer!

Het grootste gedeelte van deze data is helaas rotzooi. Denk aan twee protonen die als biljartballen van elkaar af zijn geëkaatst, maar voor de rest niets interessants hebben gedaan. Dit heet een *soft scatter interaction*, omdat de protonen elastisch tegen elkaar gebotst zijn. De inelastische botsingen heten ook wel *hard scatter interactions*, waarbij de protonen zo hard op elkaar klappen dat hun interne quarks en gluonen met elkaar de interactie aangaan. Dit is waar alle interessante natuurkunde gebeurt, zoals de productie van [Higgsbosonen](#). De uitdaging is dus het filteren van de gigantische datastromen zodat we zoveel mogelijk *hard scatter*-data behouden, terwijl de totale hoeveelheid data die we opslaan behapbaar blijft.

Het trigger-systeem

Het filteren van de detector-data wordt gedaan op basis van *triggers*. Het idee achter een trigger is het volgende: stel dat ik een muon met een hoge impuls in mijn data heb zitten, dan is er een grote kans dat er in die data iets interessants zit. Alle data met muonen met een bepaalde minimumimpuls wil ik dan dus doorlaten. Dit is een voorbeeld van een *low-level trigger*, waarbij de data gefilterd worden op basis van heel simpele criteria. Deze eerste

stap in het filteren gooit al 99,7% van de data weg. Nog eens 99% van wat overblijft wordt weggegooid door de *high-level triggers*, die eisen kunnen stellen als het voorkomen van twee of meer muonen (zie afbeelding 2), of drie b-quarks. Als de data voldoen aan geen van die verschillende eisen, worden de gegevens weggegooid.



Afbeelding 2. De brokstukken van een protonenbotsing, gemeten door de CMS-detector. De rode lijnen zijn muonen, aangezien die als enige deeltjes niet geabsorbeerd worden in de binnenste delen van de detector. Met wel vier muonen wordt deze botsing doorgelaten door zowel de low-level als high-level triggers. Credits: CERN, Thomas Mc Cauley (University of Notre Dame, US)

Nu kan de detector ons niet direct zeggen of er na de botsing sprake van een muon of een quark was. Om deze informatie te achterhalen, moeten er *reconstructie-algoritmes* worden uitgevoerd op de ruwe data. Voor de *low-level triggers* zijn dit op het moment uitgekleepte varianten van onze offline algoritmes daarvoor. Dit moet wel, omdat de ruwe data van een botsing maar 4 microseconden beschikbaar is voordat er plaats moet worden gemaakt voor nieuwe data. Deze reconstructie-algoritmes zijn *rule based*, dus gebaseerd op hoe wij natuurkundigen denken dat de data eruitziet. Denk aan een regel als: “Als je binnen deze afstand twee hits in de detector hebt, dan komen die van hetzelfde deeltje”. Dergelijke algoritmes werken allemaal prima. Dit is de manier waarop we het Higgsboson hebben gevonden, en we komen nu steeds dichterbij de detectie van processen met twee

Higgsbosonen – processen die 1000 keer minder vaak gebeuren dan de productie van een enkel Higgsboson.

Toch valt er ook nog heel veel te winnen, en dit is waar machine learning om de hoek komt kijken. Zowel CMS als ATLAS (waar ik zelf werk) zoeken momenteel naar nieuwe deeltjes die [donkere materie](#) zouden kunnen verklaren. Als zo'n deeltje bestaat, dan zou het een heel kleine kans moeten hebben om gevormd te worden in een protonenbotsing. Om die kans te vergroten gaat de LHC tussen 2026 en 2028 geüpgraded worden om van 60 botsingen per bundel naar 140 botsingen per bundel te gaan. ATLAS en CMS gaan ook uitgebreide upgrades krijgen, waardoor uiteindelijk de totale datastroom naar 5 PB/s gaat, 5% van het wereldwijde internetverkeer. Machine learning reconstructie-algoritmes zijn preciezer in het reconstrueren van de data, waardoor de filtering efficiënter wordt, en een hoger percentage van de data die in onze uiteindelijke datasets eindigen ook interessant is.

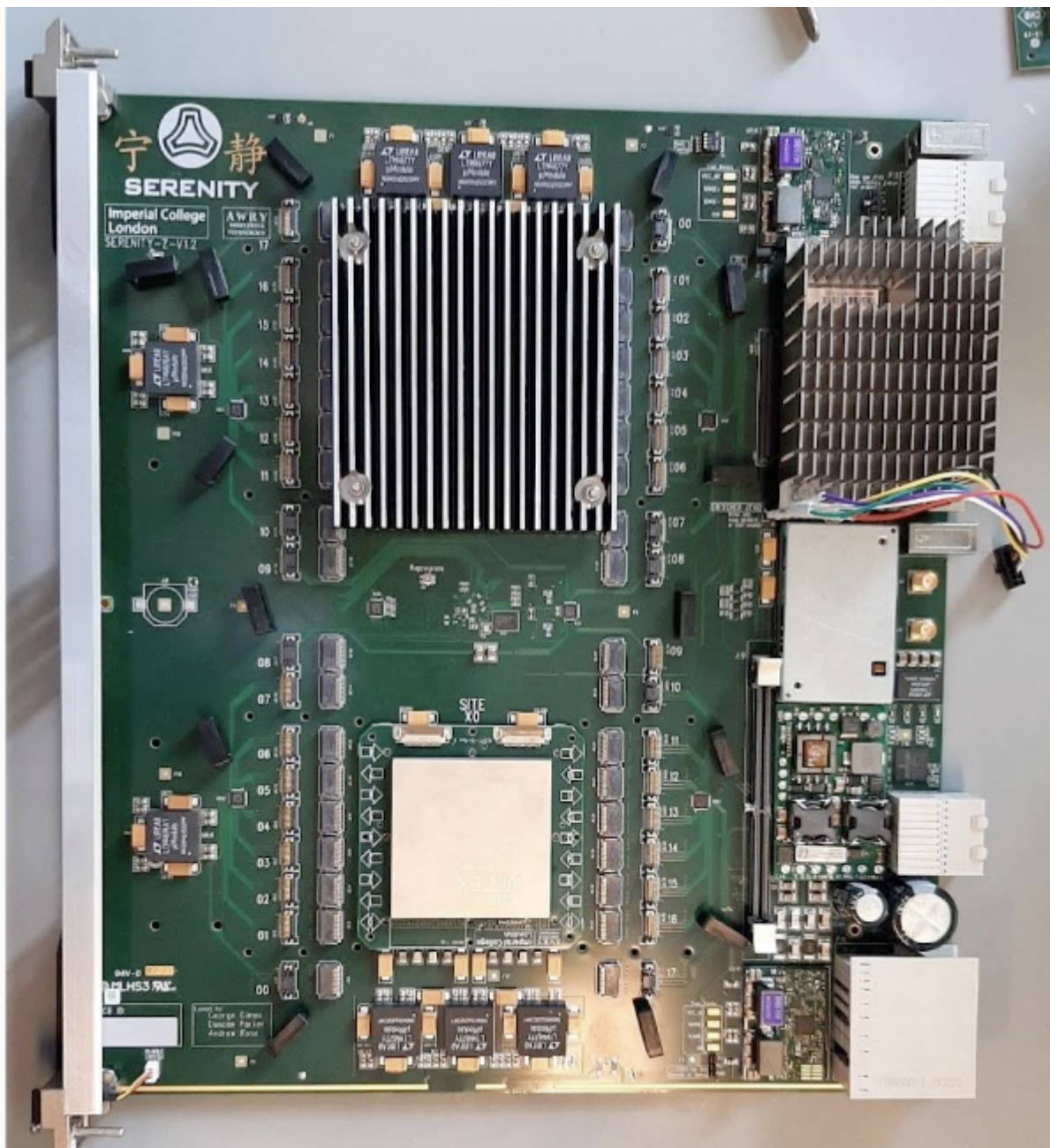
Een tweede aspect is dat donkere materie een signaal in de detector zou kunnen achterlaten wat niet opgepikt wordt door onze huidige *triggers*, omdat het niet lijkt op hoe wij denken dat interessante natuurkunde eruitziet. Misschien zijn er bij deze processen bijvoorbeeld wel helemaal geen muonen betrokken. Een specifieke klasse van machine learning-algoritmes, *anomaly detection*-algoritmes, zou dit soort data wel eruit kunnen pikken. Al met al zijn er dus voldoende redenen om machine learning-algoritmes toe te voegen aan onze *trigger*-systemen.

Ultrasnelle machine learning

Het grote probleem bij dit alles is *snelheid*. OpenAI's ChatGPT 4o kan tot 10 seconden nodig hebben om een antwoord te genereren, terwijl het programma 128 GPUs tot zijn beschikking heeft. De *low-level triggers* mogen daarentegen maar een paar microseconden de tijd nemen, en alles moet op een enkele chip gebeuren. De eerste en voor de hand liggende aanpassing die je kan maken is de grootte van het machine learning-model dat je wilt gebruiken. GPT 4o werkt naar verluidt met een biljoen (10^{12}) parameters, dus dan is het ook niet zo mal dat het even duurt voor je een antwoord krijgt. Kleinere modellen, met zo'n vijfduizend parameters, zijn automatisch een stuk sneller.

Hiermee ben je er helaas nog niet, omdat de rekentijd op een normale processor voor zo'n

model nog steeds in de orde grootte van milliseconden ligt. Er bestaat gelukkig speciale hardware die supersnel berekeningen kan uitvoeren. ATLAS en CMS gebruiken nu nog voornamelijk ASICs: *application-specific integrated circuits*, wat chips zijn die in de fabriek geprogrammeerd worden (dat wil zeggen: alle bits worden van tevoren op één of nul gezet, op basis van een stuk code dat je aanlevert). Dit soort chips zijn supersnel, maar minder handig voor machine learning-toepassingen, omdat een model dan nooit geüpdatet kan worden. Een betere optie zijn FPGAs: *field-programmable gate arrays*, die wel geüpdatet kunnen worden, en nog steeds snel genoeg zijn. Een voorbeeld van een processor met een FPGA zie je in afbeelding 3.



Afbeelding 3. Een processor-prototype voor CMS's trigger-systeem. De

FPGA is de vierkante chip in het midden van de onderste helft van de processor.

Credits: CERN, Michalis Bachtis (CERN)

FPGAs werken niet met normale code (zoals C, Python, etc.), maar met machinecode die bestaat uit ruwe bits. Het trainen van een machine learning-algoritme gebeurt echter wél in talen als C en Python, en dus er is een programma daartussen nodig dat je Pythoncode naar machinecode vertaalt. Helaas is dat geen eenduidig proces. Waar er bij gesproken talen meerdere manieren zijn om dezelfde zin van de ene taal naar de andere te vertalen, zo zijn er bij programmeertalen ook meerdere manieren om dezelfde code naar machinecode te vertalen. Maar waar in gesproken talen de connotatie misschien verandert, verandert bij machinetalen de snelheid van het programma. De commercieel beschikbare vertaalprogramma's hebben hier last van, en dus hebben deeltjesfysici het op zich genomen om zelf een vertaler te bouwen: [hls4ml](#). Met deze vertaler is het nu mogelijk om rekestijden te behalen van nanoseconden!

CMS heeft in 2023 op deze manier een anomaly detection-algoritme toegevoegd aan hun low-level trigger-systeem, en dit is naar alle waarschijnlijkheid nog maar het begin van machine learning-toepassingen in de trigger-systemen. Met alle nabije upgrades aan de LHC en de ATLAS- en CMS-detectoren, kunnen we uitkijken naar de (hopelijk) interessante stroom aan data die ons hierdoor te wachten staat.