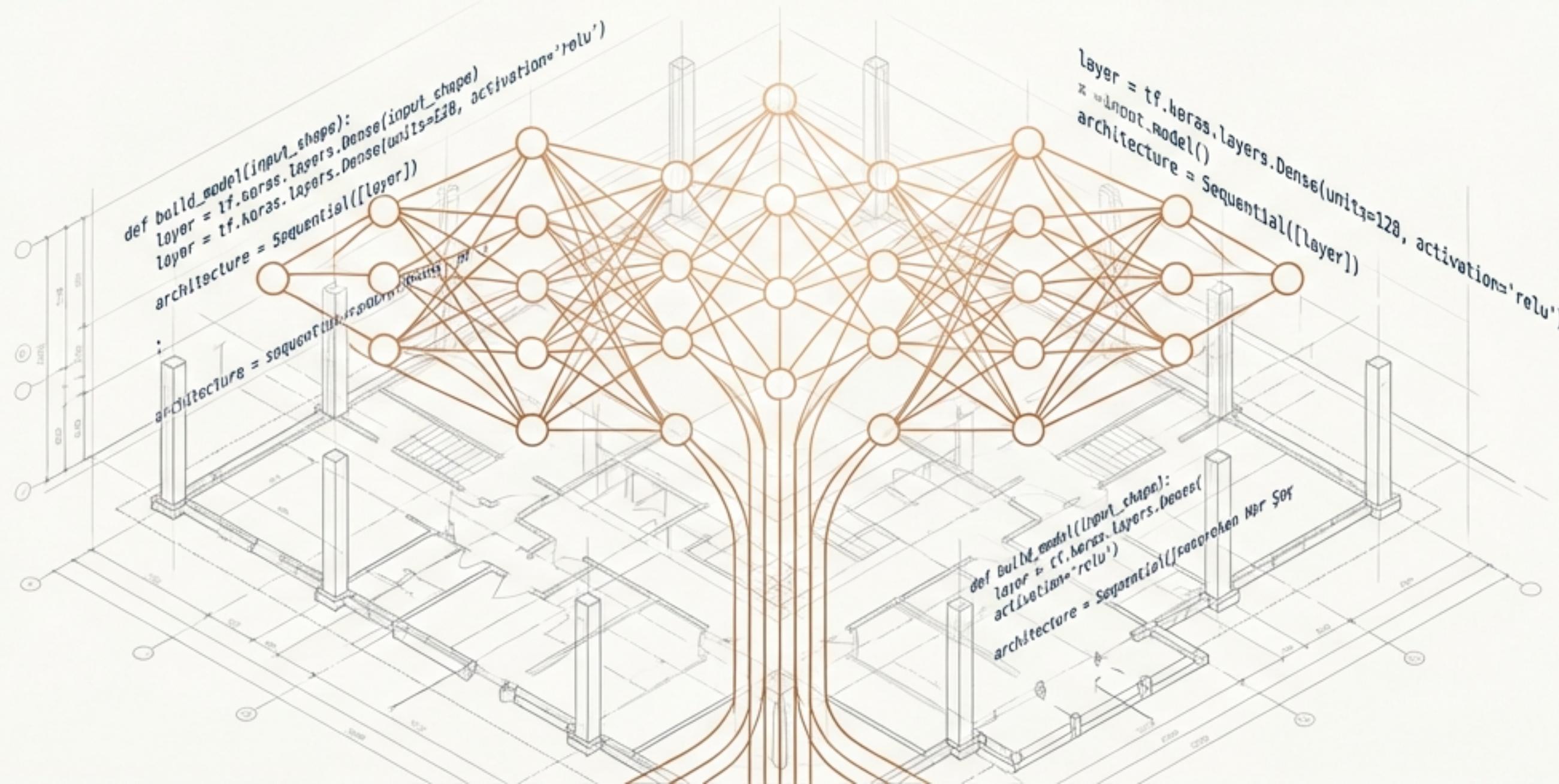


Yapay Zekanın Mimarisi: Makine Öğrenmesi Yolculuğunuzun Temel Taşları

Kavramlar, Diller, Donanımlar ve Araçlar: Başlamak İçin Bilmeniz Gereken Her Şey



Ufuk Turu: (ExtrapseBold) Artırılmış Gerçeklik ile Neler İnşa Edebiliriz?

Artırılmış gerçeklik (AR), dijital bilgileri ve nesneleri gerçek dünya üzerine yerleştirerek zenginleştirilmiş bir deneyim sunar. Bu tür karmaşık ve büyülü deneyimleri hayatı geçirmek için bir dizi güçlü araca, felsefeye ve donanıma ihtiyacımız var. Gelin, bu alet çantasını birlikte inceleyelim.



Eğitim
İnteraktif öğrenme
materyalleri



Sağlık
Cerrahi eğitim,
hasta tedavisi



Turizm
Tarihi mekanların
interaktif tanıtımı

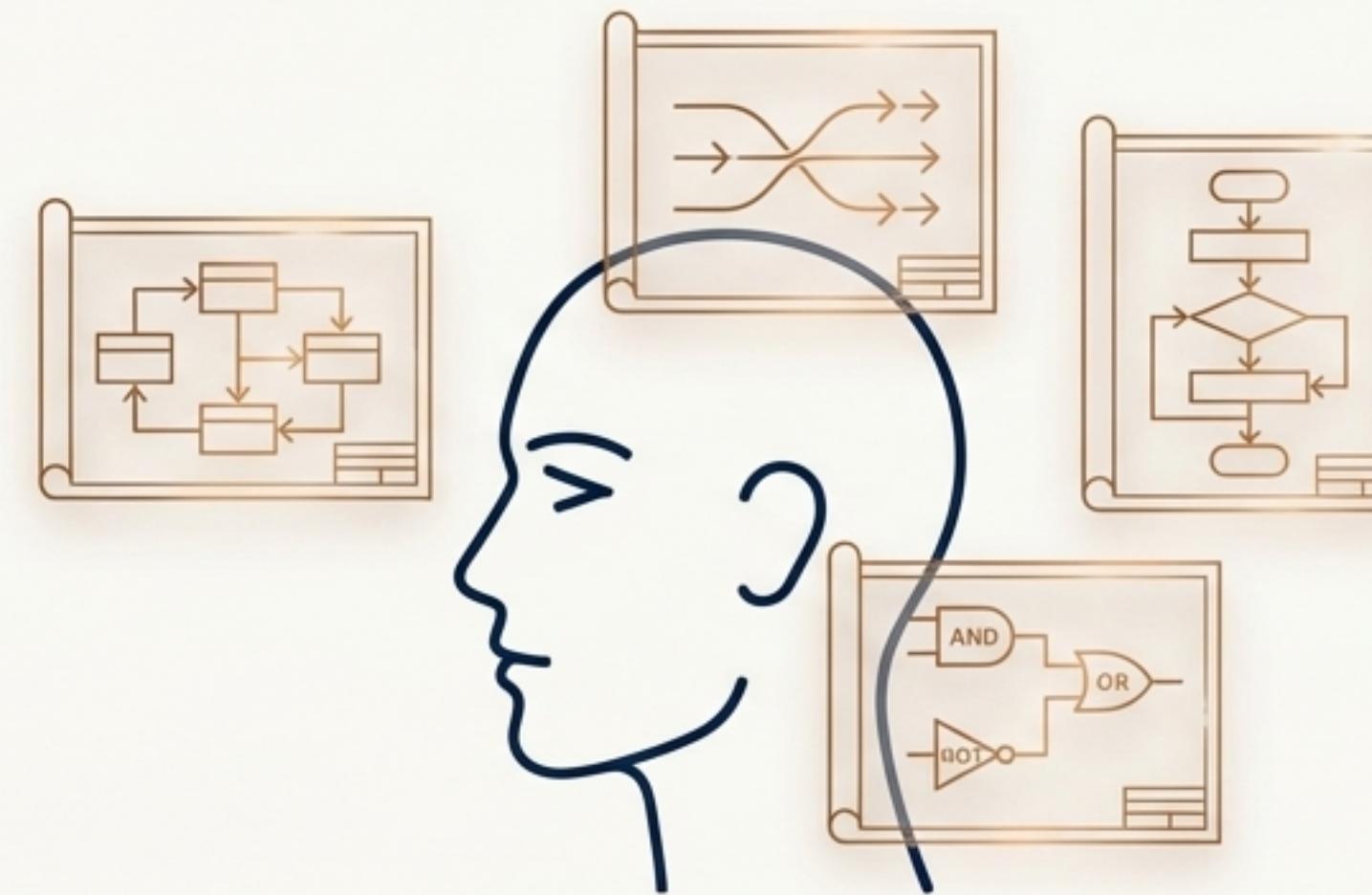


Perakende
Ürünleri sanal
olarak deneme

Mimarlık
Yapıların sanal
modelleri

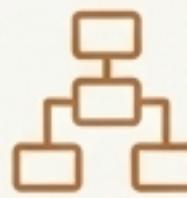
Proje Planı: Kodlamadan Önce Nasıl Düşünürüz?

Programlama Paradigmaları: Düşünce Şapkalarımız



Programlama paradigması, belirli bir yöntemle programlama yapılmasını sağlayan felsefi bir yaklaşımdır. Tıpkı bir mimarın farklı projeler için farklı stiller (modern, klasik vb.) kullanması gibi, iyi bir geliştirici de doğru problem için doğru düşünce yapısını seçer. Günümüzün güçlü dilleri genellikle 'Çok Paradigmalı'dır, yani bize birden fazla yaklaşımı kullanma esnekliği sunarlar.

Dört Temel Yaklaşım ve Mimari Stilleri



Nesne Yönelimli (OOP)

İdea: Veri ve fonksiyonları nesneler ve sınıflar içinde bir arada tutar.

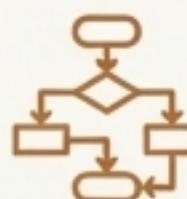
Örnenğin: Java, C++, Python



Fonksiyonel

İdea: Saf fonksiyonlar ve değişmez veriler kullanır, yan etkilerden kaçınır.

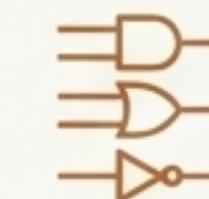
İnter: Haskell, Scala, LISP, Inter



Prosedürel

İdea: Kod, fonksiyonlar ve prosedürler kullanılarak adım adım düzenlenir.

C, Pascal, Fortran, Inter



Mantık

İdea: Problemi çözmek için kurallar ve mantıksal ifadeler kullanılır.

Prolog, PLC, Inter

Malzemelerimiz: Doğru İşe Doğru Alet

Senaryo: Robotik Sistem Geliştirme

Python



Avantajlar: Yazım Kolaylığı, Hızlı Geliştirme. Hızlı prototipleme ve deney için ideal.

Dezavantaj: Anlık tepki süresi kritik olduğunda performans yetersiz kalabilir.

C++



Avantajlar: Ham Performans, Donanım Hakimiyeti. Hızlı tepki süresi gerektiren görevler için mükemmel.

Dezavantaj: Yazımı ve yönetimi daha karmaşıktır.

En iyi dil diye bir şey yoktur; sadece görev için en uygun dil vardır.

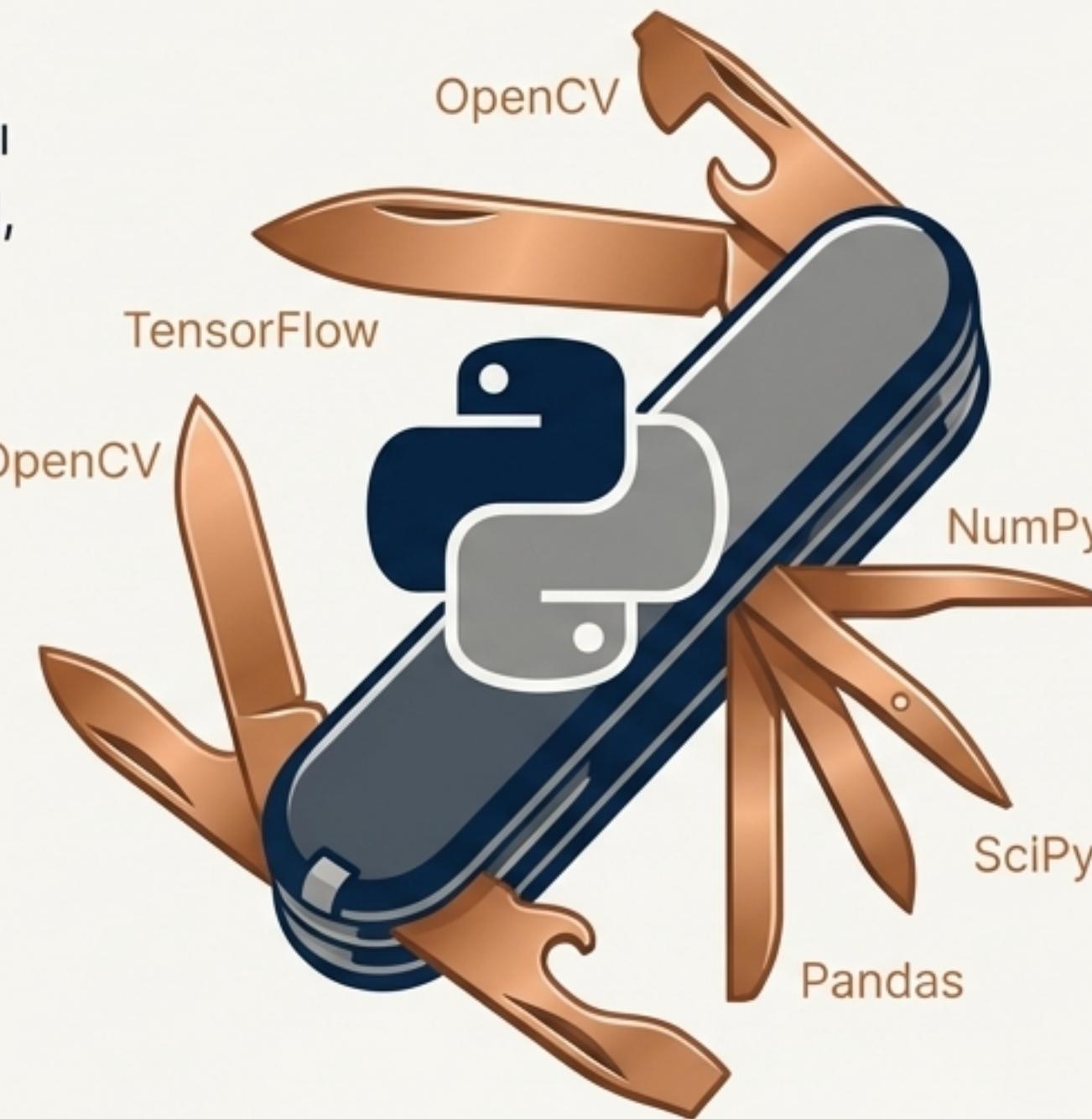
Ana Aracımız Python: Yapay Zeka'nın ‘İsviçre Çakısı’

Yolculuğumuzda bize en çok eşlik edecek dil Python. Çok paradigmalı yapısı (Nesne Yönelimli, Prosedürel, Fonksiyonel) sayesinde bize inanılmaz bir esneklik sunar.

Güçlü Yanları

Devasa Kütüphane Ekosistemi:
OpenCV, SciPy, TensorFlow,
NumPy gibi hazır araçlarla dolu.

Büyük Bulut Desteği: Google
Cloud, Amazon Web Services ve
Microsoft Azure tarafından tam
desteklenir.



Geliştirilmesi Gereken Yönleri

Performans: C++ gibi dillere göre çalışma hızı daha yavaş olabilir.

Mobil Destek: Mobil ortamlar için desteği daha zayıftır.

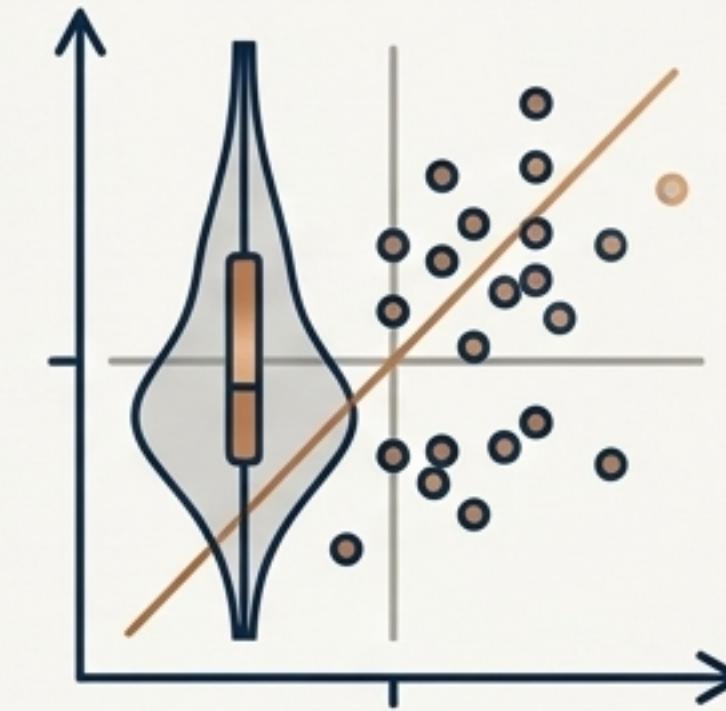
Uzmanlaşmış Aletler: C++ ve R



C++ - Performans Canavarı

Donanım üzerinde oldukça yüksek bir hakimiyeti vardır. Çalışma zamanı ve performans ile ön plana çıkar. Inter.

Kullanım Alanları
Oyun motorları, robotik sistemler, yüksek hız gerektiren yapay zeka modülleri. Inter



R - İstatistikçinin Neşteri

İstatistiksel hesaplamalar, veri analizi ve görselleştirme için tasarlanmış bir dil ve ortamdır. Inter.

Güçlü Yanı: Grafikleri yayın kalitesindedir. Python'a göre görselleştirmede başka bir seviyededir. Inter.

Zayıf Yanı: Python'a göre daha yavaş çalışabilir. Inter.

Ufukta Başka Neler Var? Diğer Önemli Diller

LISP

Yapay Zekanın Kökleri

AI araştırmaları için
doğmuş, sembolik işlemler
ve rekürsiyon üzerine kurulu
fonksiyonel bir dil.

Haskell

Saf Fonksiyonel

Matematiksel ve modelleme
programlamada en başarılı
örneklerden biri.

Julia

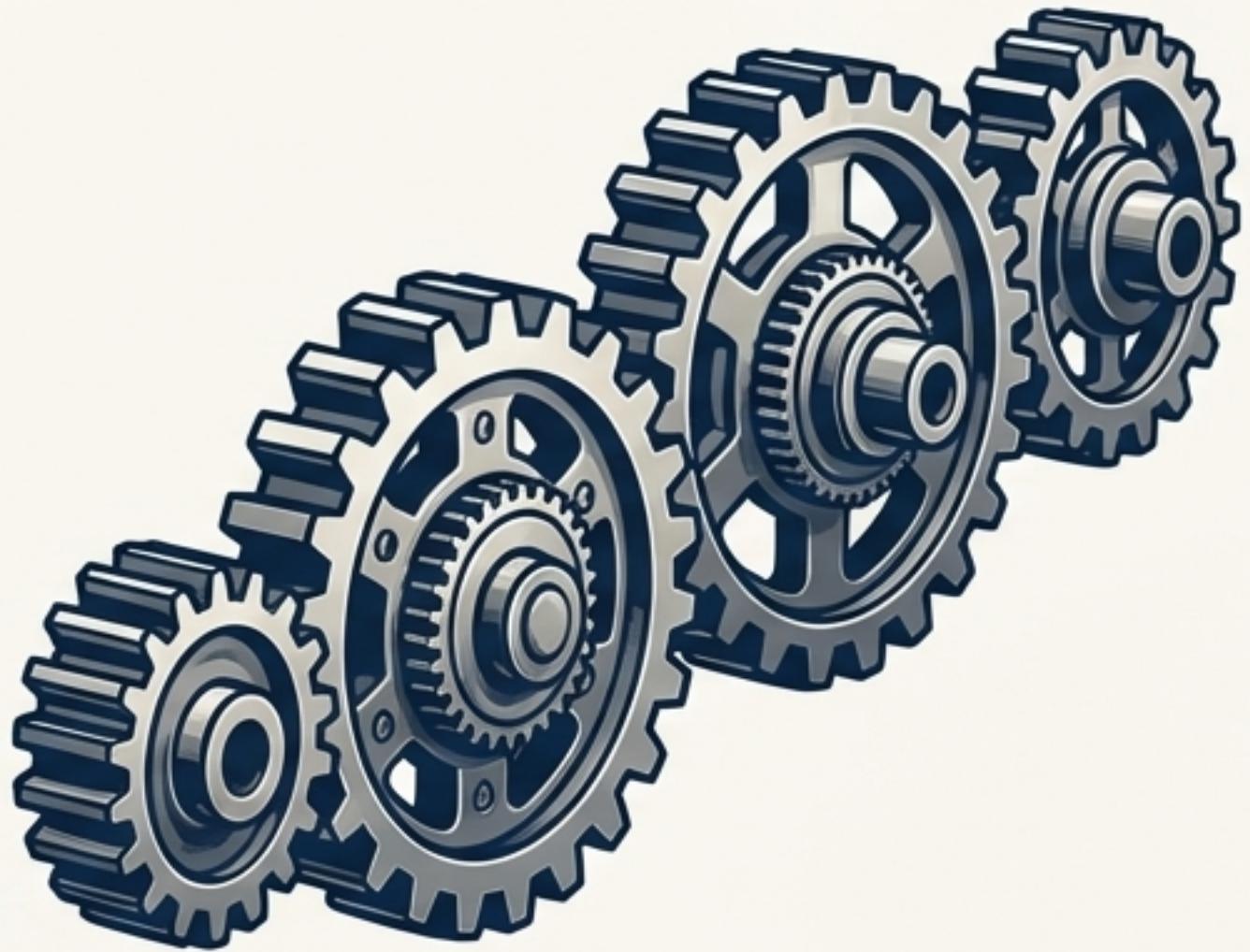
Yeni Nesil Güç

'Python kadar kolay, C++
kadar güçlü' sloganıyla yola
çıktı. Compile ile çalışır,
yüksek performans sunar.

Motor Odası: Kodumuz Nerede Çalışıyor?

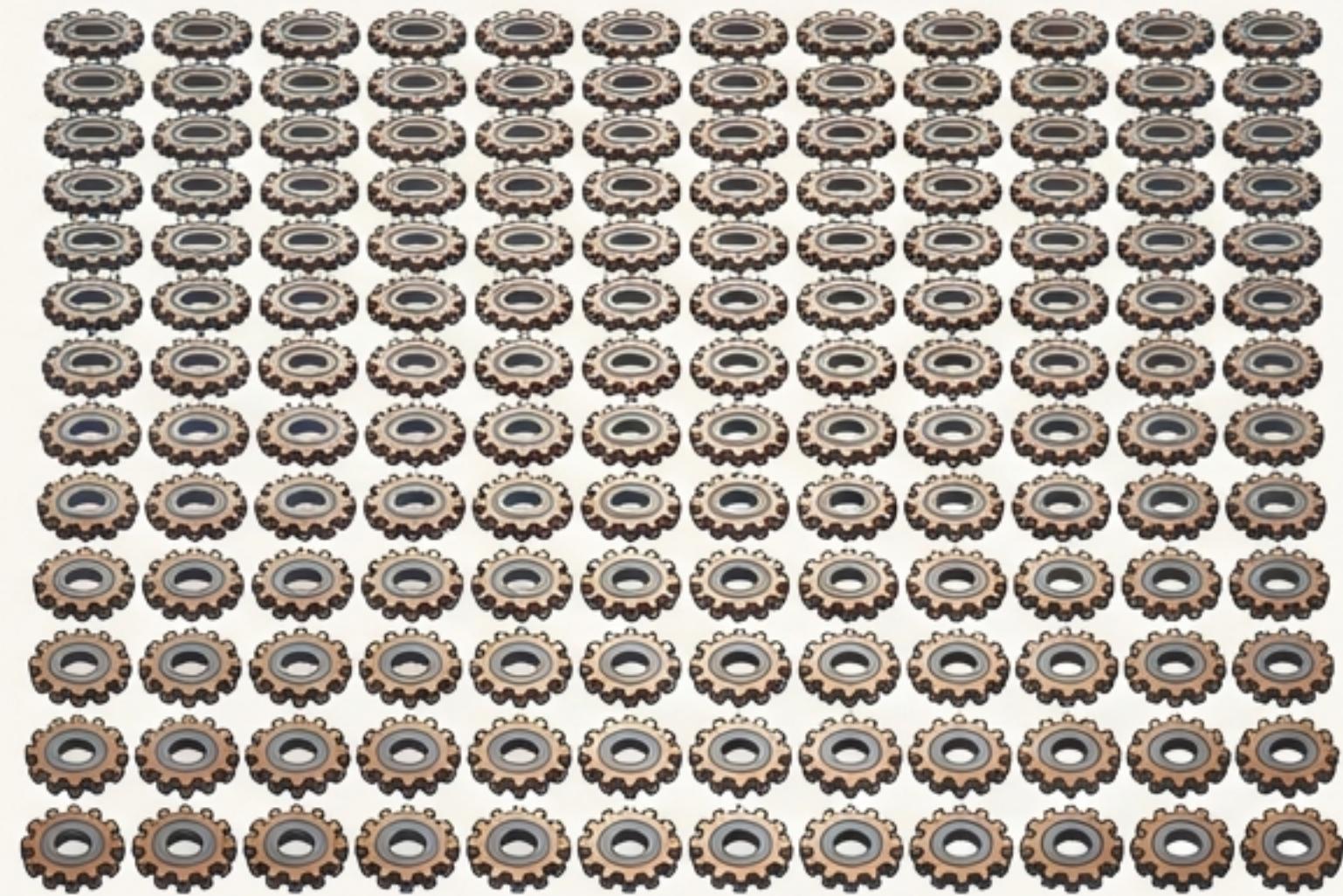
Neden Yapay Zeka için Donanım Seçimi Kritik?

CPU: Birkaç Dahi Uzman



Daha az sayıda ama çok güçlü çekirdeğe sahiptir. Karmaşık ve seri (ardışık) görevler için optimize edilmiştir.

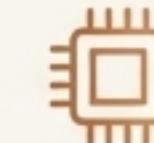
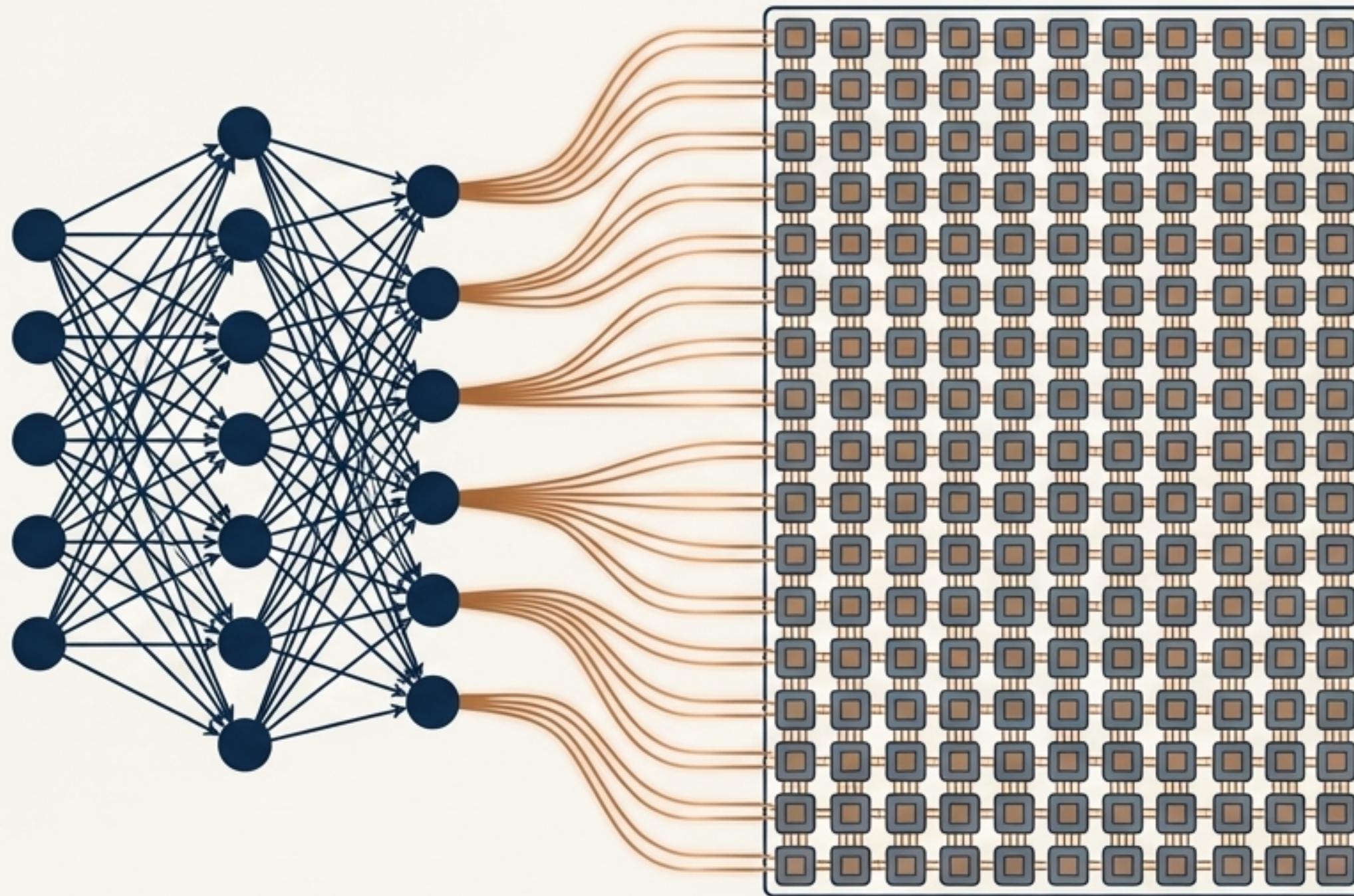
GPU: Binlerce Çalışkan İşçiden Oluşan bir Ordu



Daha yavaş olmalarına rağmen binlerce çekirdeğe sahiptir. Aynı anda binlerce basit görevi yapabilirler (paralel işleme).

Yapay Zeka Neden GPU'yu Seviyor?

Yapay sinir ağlarının eğitimi, doğası gereği devasa bir paralel problemdir. Binlerce hesaplamanın aynı anda yapılması gereklidir. Bu yüzden GPU'lar bu iş için mükemmeldir.



Paralel İşleme Yeteneği

Binlerce çekirdek, sinir ağındaki binlerce işlemi aynı anda gerçekleştirir.



Yüksek Hesaplama Kapasitesi

Büyük matris ve vektör işlemleri için idealdir.



Hız ve Verimlilik

Eğitim sürecini haftalardan saatlere indirebilir.

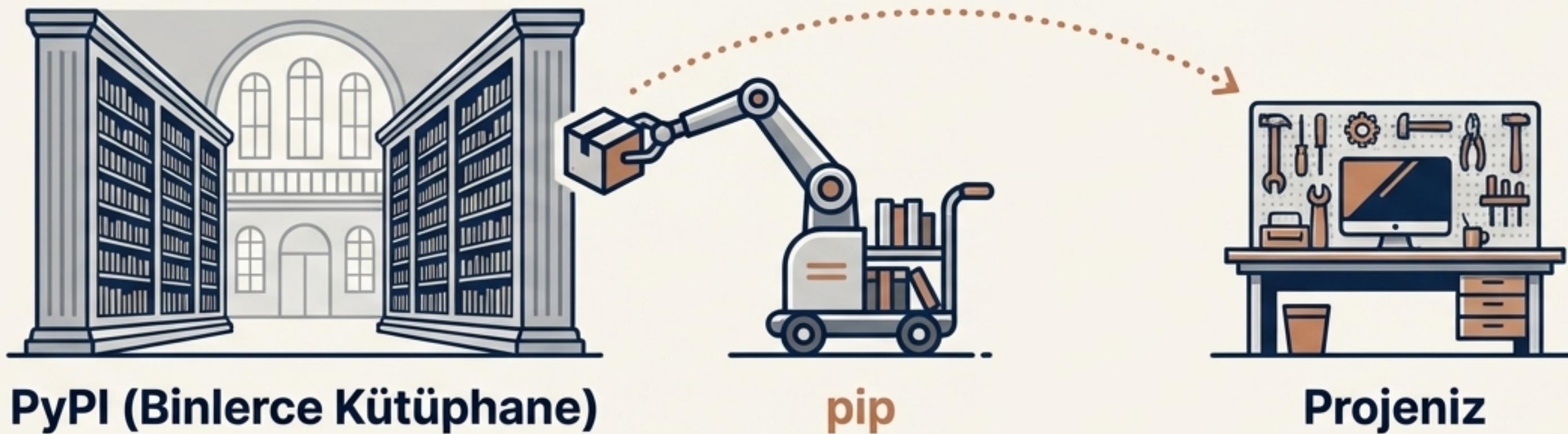


Optimize Edilmiş Kütüphaneler

TensorFlow ve PyTorch gibi kütüphaneler, GPU'ların gücünü sonuna kadar kullanmak için tasarlanmıştır.

Atölye Düzeni: Araçlarımızı Nasıl Yönetiriz?

Python'ın en büyük gücü, PyPI (Python Package Index) adı verilen devasa bir kütüphane deposundan gelir. Peki bu depodan istediğimiz aracı (kütüphaneyi) projemize nasıl getiririz?



PyPI (Binlerce Kütüphane)

pip

Projeniz

Cevap: `pip` (Pip Installs Packages)

Analoji: PyPI, Python için dev bir halk kütüphanesi gibidir. `pip` ise o kütüphaneden istediğiniz kitabı yanında size getiren süper hızlı kütüphanecidir.

Ne Yapar?: Python kütüphanelerini ve bağımlılıklarını kolayca yüklemeyi, güncellemeyi ve yönetmeyi sağlar.

`pip` Komutları: Atölyenin Temel Kontrolleri

Paket Yükleme

```
# NumPy kütüphanesini yükler  
pip install numpy
```

Paket Kaldırma

```
# NumPy kütüphanesini kaldırır  
pip uninstall numpy
```

Paket Güncelleme

```
# NumPy kütüphanesini en son sürümeye günceller  
pip install --upgrade numpy
```

Proje Bağımlılıklarını Yükleme

```
# requirements.txt dosyasındaki tüm paketleri tek seferde yükler  
pip install -r requirements.txt
```

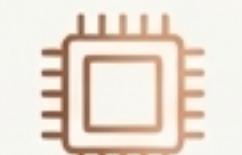
Paket Yükleme

```
# NumPy kütüphanesini kaldırır  
pip install numpy
```

Yüklü Paketleri Listeleme

```
# Ortamınızdaki tüm paketleri listeler  
pip list
```

Mimarının Bütün Parçaları: Bir Fikirden Gerçeğe

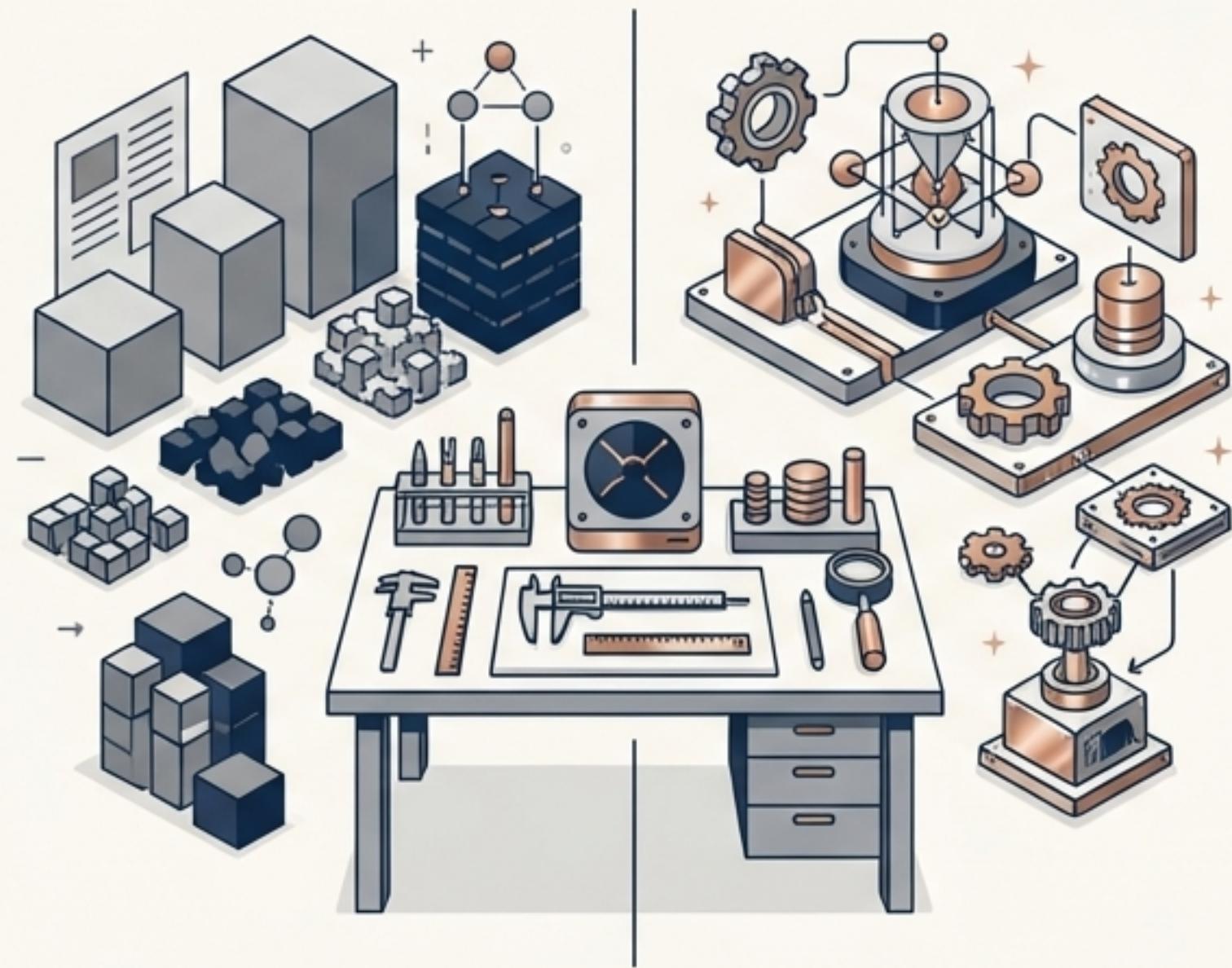
- 1  **VİZYON:** Bir Artırılmış Gerçeklik Uygulaması geliştirmek istiyoruz.
- 2  **PROJE PLANI:** Nesne Yönelimli Paradigmayı (OOP) seçiyoruz.
- 3  **MALZEME:** Bunu **Python** dili ile hayata geçiriyoruz.
- 4  **ARAÇLAR:** Güçlü **TensorFlow** kütüphanesini '**pip**' ile yükliyoruz.
- 5  **MOTOR ODASI:** Bu ağır hesaplamaları bir **GPU** üzerinde çalıştırıyoruz.

Gördüğünüz gibi, tartıştığımız her kavram, büyük bir hedefi gerçekleştirmek için birlikte çalışan bir sistemin vazgeçilmez bir parçasıdır.

Yolculuğunuzun Bir Sonraki Durağı

Artık yapay zeka projelerinin temel mimarisini ve araçlarını anladığımıza göre, atölyeye girip malzemeleri işlemeye başlama zamanı. Önümüzdeki derslerde bu temel üzerine inşa edeceğiz.

- Sıradaki Konular
- Vektörler ve Matrisler: Verinin Dili
- NumPy: Sayısal Hesaplamanın Temeli
- Pandas: Veri Manipülasyon Sanatı
- Matplotlib & Seaborn: Veriyi Görselleştirme
- Ve ardından... Makine Öğrenmesi ve Derin Öğrenme Modelleri!



Teşekkürler

Sorularınız?
