

خطة مشروع متكاملة: GeoSynth AI

عنوان المشروع: GeoSynth AI؛ نموذج توليدی متقدم لتحويل الأوصاف النصية إلى صور هندسية عالية الدقة.

الرؤية (Vision): أن تكون الأداة الرائدة للمصممين، والمهندسين، والمعلمين لإنشاء تمثيلات بصرية دقيقة للأشكال الهندسية المعقدة والمركبة من خلال أوامر لغوية بسيطة، مما يسرّع من وتبيرة الإبداع والتوثيق التقني.

المرحلة الأولى: التخطيط وتحديد المتطلبات (Planning & Scoping)

هذه المرحلة تسبق التنفيذ وتتضمن أننا نبني المنتج الصحيح.

1. تحديد نطاق الأشكال الهندسية:

- * المستوى الأساسي (MVP - Minimum Viable Product): أشكال ثنائية الأبعاد بسيطة (دائرة، مربع، مثلث، مستطيل، خط، نجمة).
- * المستوى المتقدم: أشكال معقدة (مضلعات منتظمة وغير منتظمة، أشكال إهليجية)، تركيبات (شكلين أو أكثر في صورة واحدة)، أشكال ثلاثية الأبعاد بسيطة (مكعب، كرة، هرم).
- * مستوى الخبراء: أشكال فrac{als} (Fractals) مثل مجموعة ماندلبروت، أو منحنيات رياضية معقدة.
- * قرار: سنبدأ بالمستوى الأساسي والمتقدم معاً، مع التركيز على التركيبات.

- .2. تحديد خصائص الصورة (Image Attributes):
 - * الألوان: دعم أسماء الألوان ("أحمر") وقيم RGB ((255, 0, 0)).
 - * الخلفية: لون خلفية سادة، أو خلفية شفافة (PNG).
 - * الحجم والموقع: التحكم في الحجم النسبي ("دائرة كبيرة") والموضع ("مربع في الزاوية العلوية اليسرى").

- * الحدود (Stroke): لون وسمك حدود الشكل.
- * الشفافية (Opacity): التحكم في درجة شفافية الشكل.

3. تحديد مواصفات المخرجات:
 - * دقة الصورة: نبدأ بـ 512\$ \ 512\$ times بكسł، مع إمكانية الترقية إلى 1024\$ \ 1024\$ JPEG.
 - * صيغ الملفات: PNG (لدعم الشفافية).

المرحلة الثانية: بناء خط أنابيب البيانات (Data Pipeline)

هذه هي أهم مرحلة وتحدد جودة النموذج النهائية.
"البيانات هي النفط الجديد".

1. توليد البيانات الأصلية (Programmatic Data Generation)

بدلاً من جمع الصور يدوياً، سنقوم بإنشائها برمجياً للتحكم الكامل.

* الأدوات: سنستخدم مكتبات مثل Pillow أو Python أو Cairo في OpenCV

* المنهجية:

1. إنشاء سكريبت (Script) يقوم بتوليد الصور بناءً على مجموعة من المعلمات العشوائية (Parameters) ضمن النطاق الذي حددناه في المرحلة الأولى.

2. لكل صورة يتم توليدها، يجب حفظ ملف بيانات وصفية (Metadata) بصيغة JSON. هذا الملف هو "شهادة ميلاد" الصورة ويحتوي على كل شيء عنها.

* مثال لملف `image_0001.json`

```
{  
    "filename": "image_0001.png",  
    "resolution": [512, 512],  
    "background_color": "white",  
    "objects": [  
        {
```

```
        "shape": "circle",
        "color": "red",
        "center": [256, 128],
        "radius": 50,
        "border_color": "black",
        "border_width": 2
    },
    {
        "shape": "square",
        "color": "blue",
        "top_left": [150, 300],
        "size": 80
    }
]
```

* حجم البيانات: نهدف إلى توليد 10,000 إلى 50,000 صورة فريدة مع ملفات JSON الخاصة بها. الجودة والتحكم هنا أهم من الكمية الهائلة.

(Advanced Text Description Generation)

سنستخدم ملفات JSON لتوليد أوصاف نصية غنية ومتعددة.

* المنهجية:

1. **قوالب نصية (Templates):** إنشاء مجموعة كبيرة من القوالب اللغوية.
2. **التنوع اللغوي:** استخدام مرادفات (e.g., "دائرة حمراء كبيرة"، "كرة بلون أحمر ضخمة").
3. **وصف العلاقات:** توليد أوصاف للعلاقات بين الأشكال ("دائرة حمراء فوق مربع أزرق"، "مثلث أخضر إلى يسار مستطيل أصفر").
4. **السكريبت:** برمجة سكريبت يقرأ كل ملف JSON ويقوم بتوليد 3-5 أوصاف نصية مختلفة لكل صورة لزيادة قوة تدريب النموذج.

* مثال للمخرجات:

* الصورة image_0001.png سبقت بها نصوص مثل:

* "A red circle with a black border above a blue square on a white

background."

- * "A large blue square is positioned below a smaller red circle."
- * "An image containing two shapes: a red circle and a blue square."

3. تعزيز البيانات (Data) ##### (Augmentation)

هنا سنزيد من قوة بياناتنا لمقاومة الأخطاء.

- * تعزيز الصور (Image Augmentation):
- * تطبيق تحويلات طفيفة جدًا لا تؤثر على الشكل الهندسي الأساسي.
- * الأدوات: مكتبة .Albumentations
- * التقنيات: إضافة ضوضاء خفيفة (Gaussian)، تغيير طفيف في السطوع والتبابين (Noise)، تمويه بسيط جدًا (Brightness/Contrast)، (Blur).
- * تعزيز النصوص (Text Augmentation):
- * تقنية متقدمة: الترجمة العكسية (Back-Translation)

نترجم الوصف من العربية إلى الإنجليزية ثم إلى الفرنسية ثم نعيده للعربية باستخدام APIs للحصول على صيغ لغوية جديدة ومختلفة.

المرحلة الثالثة: تطوير وتدريب النموذج (Model Development)

هنا يبدأ سحر الذكاء الاصطناعي.

4. اختيار وتحميل نموذج-  Text-to-Image #####

* النموذج الأساسي (Base Model) : Stable Diffusion (SD 1.5 أو SDXL). السبب: نماذج قوية، مفتوحة المصدر، ومدعومة بشكل هائل من المجتمع.

* المكتبات: سنعتمد على مكتبة diffusers، وهي المعيار الصناعي لهذه المهمة. Hugging Face

* تقنية التحكم الدقيق (- Precision Control):
اختياري لكن موصى به):

* **ControlNet**: سنفكر في تدريب نموذج "إضافي" يستطيع أخذ "خرسفة حواضن" (Edge Map) للشكل الهندسي كمدخل إضافي. هذا سيمنح النموذج قدرة خارقة على الالتزام بحدود الأشكال بدقة متناهية.

Fine-5. الضبط الدقيق للنموذج (Tuning)

لن نقوم بتدريب النموذج من الصفر، بل سنقوم بتكييفه على بياناتنا.

- * التقنية: **.LoRA (Low-Rank Adaptation)**:
هذه هي أفضل تقنية حالياً للمهام المماثلة.
- * لماذا LoRA؟ لأنها سريعة جداً، تتطلب موارد حسابية (VRAM) أقل بكثير من التدريب الكامل، وتنتج ملفات صغيرة جداً (بضع ميغابايت) يمكن تحميلها فوق النموذج الأصلي، مما يسهل التوزيع والتجربة.

- * **البيئة التدريبية:**
- * **الأجهزة:** استخدام بطاقات رسومية (GPU) مثل NVIDIA RTX 3090/4090 أو خدمات سحابية مثل Google Colab Pro, AWS SageMaker, .vast.ai, أو
- * **الهاiperبارامترز (Hyperparameters):**
- * learning_rate: معدل التعلم (مهم جداً،) مثل 1\$-4\$.
 - * batch_size: حجم الدفعة.
- * num_train_epochs: عدد دورات التدريب.
 - * optimizer: سنستخدم AdamW.
- * سنقوم بتجربة قيم مختلفة للوصول إلى أفضل نتيجة.

Evaluation ## المرحلة الرابعة: التقييم والنشر (& Deployment

كيف نعرف أننا نجحنا وكيف نجعله متاحاً للعالم؟

* التقييم الكمي (Quantitative Evaluation):

* CLIP Score: مقياس يقيس مدى تطابق النص مع الصورة المولدة. كلما ارتفعت النتيجة، كان ذلك أفضل. سنقوم بحساب متوسط على مجموعة اختبار لم نرها أثناء التدريب.

* FID (Fréchet Inception Distance): يقيس جودة وتنوع الصور المولدة مقارنة بالصور الحقيقة.

* التقييم النوعي (Qualitative Evaluation):

* الاختبار البشري (Human Evaluation): إنشاء استبيان بسيط نعرض فيه على المستخدمين نصاً وصورةً مولدة ونطلب منهم تقييم الدقة والجودة من 1 إلى 5.

* تحليل الأخطاء (Error Analysis): إنشاء "مصفوفة إخفاق". هل النموذج يخطئ في الألوان؟ أم في المواقع النسبية؟ أم في عدد الأشكال؟ هذا التحليل يوجهنا لتحسين البيانات أو عملية التدريب.

(Packaging & Deployment)

- * التحريم:
- * حفظ أوزان LoRA المدربة (ملف safetensors).
- * إنشاء بطاقة نموذج (Model Card) مفصلة على Hugging Face تشرح كيفية استخدام النموذج، وقيوده، والبيانات التي تدرب عليها.
- * النشر (Containerization):
 - * استخدام Docker لإنشاء حاوية Container) تحتوي على كل المكتبات والملفات اللازمة لتشغيل النموذج. هذا يضمن أن النموذج سيعمل في أي بيئة بنفس الطريقة.
 - * كتابة ملف Dockerfile احترافي.

8. تصميم واجهة Gradio على Hugging Face Spaces

- * الواجهة: سنبني واجهة باستخدام Gradio لأنها بسيطة وقوية.
- * مميزات الواجهة المتقدمة:

1. حقل إدخال النص الأساسي (Text) .(Prompt

2. أمثلة جاهزة (Examples): أزرار تحتوي على أوامر معقدة يمكن للمستخدم تجربتها بنقرة واحدة.

3. خيارات متقدمة (Advanced Options) منزلقات (Sliders) للتحكم في:

* Guidance Scale (CFG): مدى التزام النموذج بالنص.

* Inference Steps: عدد خطوات التوليد .(توازن بين السرعة والجودة)

* Seed: لتوليد نفس الصورة مرة أخرى.

4. زر الإبلاغ عن خطأ (Flagging): السماح للمستخدمين بالإبلاغ عن النتائج السيئة. هذه البيانات لا تقدر بثمن لتحسين النموذج في الإصدارات القادمة.

ملخص المكتس التقني (Tech Stack)

- * لغة البرمجة: Python 3.9 +
- * توليد البيانات: Pillow, OpenCV
- * تدريب النموذج: PyTorch, Hugging Face (Diffusers, Transformers, Accelerate)
- * الواجهة: Gradio
- * النشر: Docker, Hugging Face Hub (Model & Space)
- * إدارة المشروع: Git, GitHub

خلاصة ونصيحة الخبير

هذا الهيكل يحول فكرتك من مجرد مشروع تجريبي إلى أساس لمنتج احترافي. أهم نصيحة أقدمها لك هي التركيز الشديد على جودة خط أنابيب البيانات (المرحلة الثانية). إن قضاء 80% من وقتك في توليد بيانات نظيفة، متنوعة، مع بيانات وصفية دقيقة سيجعل عملية تدريب النموذج وتقديره أسهل بعشر مرات وسينتج عنه نموذج فائق الدقة.

بالنوفيق في مشروع AI! لديك الآن خارطة طريق قوية. انطلق!

