Let W be the range of numbers, and assume 3-sum for n numbers can be solved in $f(n)$.
1. $O(n^2)$ by sorting+monotone pointers.
2. We can reduce the decision version of this problem (whether there exist 3-sum in range $[l, r]$) to $O(\log W)$ exact 3-sum calls [2, Theorem 1 (Shrinking intervals)], replace the n in the proof by 3. To solve this problem, we need $O(\log W)$ exponential searches, so the total running time is $O(\log^2 W \cdot f(n))$ (can probably combine the two steps and shave a log).
3. the problem can be reduced to the (batched) $A+B$ searching problem, which can be solved by [1, Lemma 3.3] (verify the reduction in that lemma also works for this problem), in deterministic $O((n^2/\log^2 n)(\log\log n)^{O(1)})$ (the same time bound in the paper for 3-sum).
4. when U is small we can use FFT. $O(U \log U)$.

lower bound: $\Omega(f(n))$, because this problem is 3sum-hard. see 015. 3Sum.

References

[1] Timothy M. Chan. More logarithmic-factor speedups for 3sum,(median,+)-convolution, and some geometric 3sum-hard problems. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 881–897. SIAM, 2018.

[2] Jesper Nederlof, Erik Jan van Leeuwen, and Ruben van der Zwaan. Reducing a target interval to a few exact queries. In *International Symposium on Mathematical Foundations of Computer Science*, pages 718–727. Springer, 2012.